RP04/5/6

MECHANICAL & READ/WRITE

MD-11-DZRJA-B

TESTS

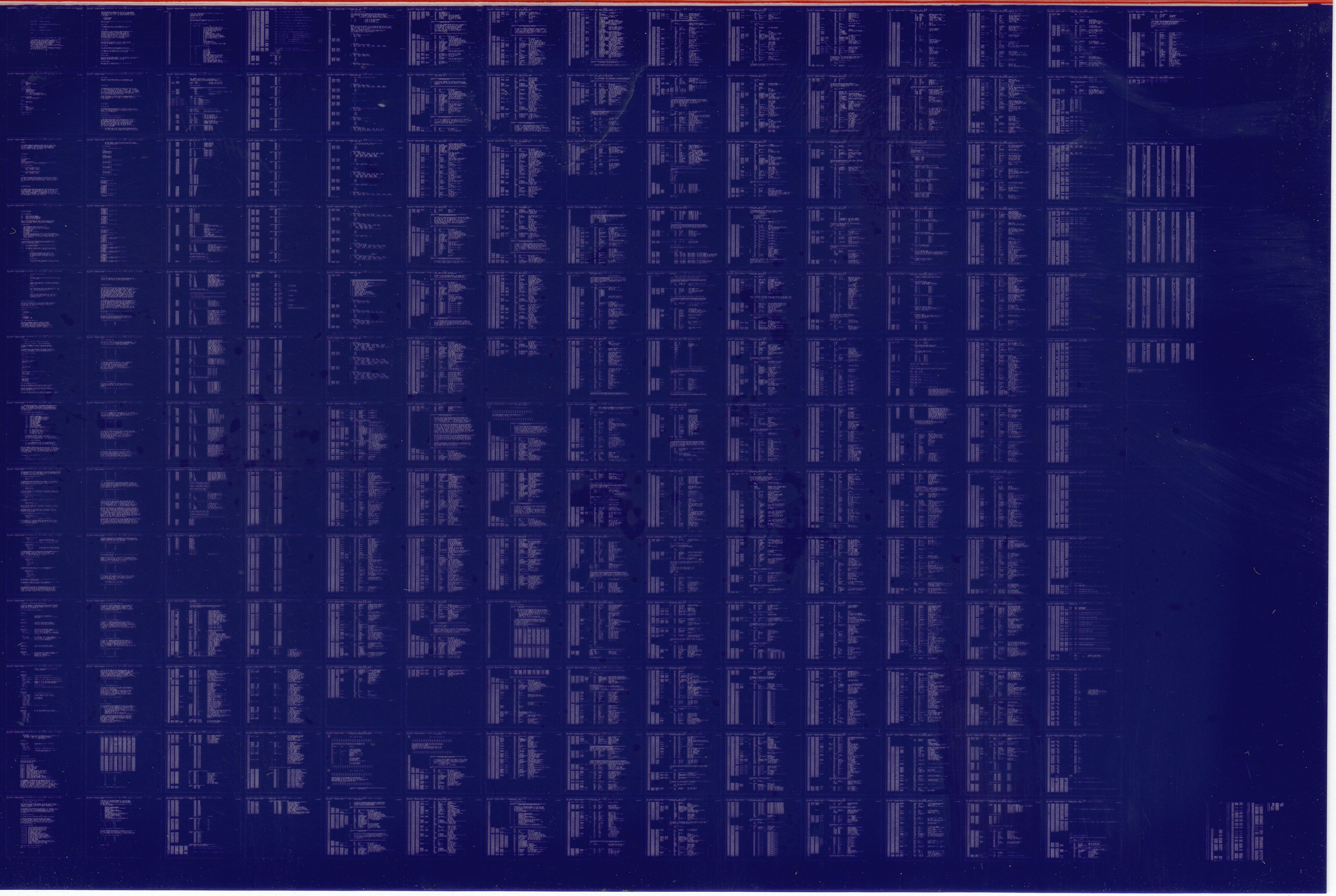
EP-DZRJA-B-DL

FICHE 1 OF 1

DEC 1977

digital

MADE IN USA



.REM @

I D E N T I F I C A T I O N

PRODUCT CODE: MAINDEC-11-DZRJA-B-D

PRODUCT NAME: RP04/5/6 MECHANICAL AND READ-WRITE TEST

DATE: SEPTEMBER 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C HESS

CONTENTS

1 ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT WILL VERIFY THAT THE DISK IS CAPABLE OF PERFORMING SEEKS, THAT THE ACCESS TIMES ARE WITHIN TOLERANCE, THAT THE TRACK AND SECTOR ADDRESSING CIRCUITRY OPERATES PROPERLY, AND THAT THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONING.

2 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K MEMORY
TELETYPE
PROGRAM LOADING DEVICE
KW11-L OR KW11-P (THE KW11-P IS REQUIRED FOR THE TIMING TESTS)
RH11 OR RH70 WITH 1 - 8 RP04/5/6 DISK DRIVES

2.2 PRELIMINARY PROGRAMS

RP04/5/6 DISKLESS CONTROLLER TEST
  PART 1 (MAINDEC-11-DZRJG)
  PART 2 (MAINDEC-11-DZRJH)

RP04/5/6 FUNCTIONAL CONTROLLER TEST
  PART 1 (MAINDEC-11-DZRJI)
  PART 2 (MAINDEC-11-DZRJJ)

2.3 MEDIA

THE PROGRAM REQUIRES THAT EACH DRIVE TO BE TESTED HAS A FORMATTED DISK PACK. THE PACK MAY BE FORMATTED IN EITHER 16-BIT OR 18-BIT MODE, DEPENDING ON THE TESTING REQUIREMENTS NOTE THAT THE PROGRAM WILL NOT TEST A MIXTURE OF DRIVES WITH BOTH 16 AND 18 BIT MODE PACKS

3 LOADING PROCEDURE

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCAITED LOADER THE PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN IF THE PROGRAM IS BEING RUN ON A PROCESSOR WITH 16K, THE 'XXDP' LOADER WILL NOT BE PRESERVED THE PROGRAM MUST BE RUN ON A SYSTEM WITH 20K OR MORE TO PRESERVE THE 'XXDP' LOADER THE 'ABSOLUTE' LOADER WILL BE PRESERVED IN A 16K SYSTEM, HOWEVER

4 STARTING PROCEDURE

4 1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS
204 SELECT OPERATING PARAMETERS
210 SELECT RH11-RH70 ADDRESSES
214 COMBINATION OF 204 AND 210

NOTE STARTING ADDRESSES 210 AND 214 ARE AVAILABLE WHEN THE PROGRAM IS INITIALLY STARTED; THESE STARTING ADDRESSES ARE TREATED AS ADDRESSES 200 OR 204 RESPECTIVELY ON RESTARTS

4 2 OPERATOR ACTION

1 LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2 LOAD A FORMATTED PACK INTO DRIVE(S) TO BE TESTED
3 BRING DRIVE(S) TO ONLINE STATE, WRITE ENABLED, AND LOCKED ON PORT
4. LOAD ADDRESS 200
5 SET SWITCHES (SEE SECTION 5)
6 PRESS START.
7 THE PROGRAM WILL TYPEOUT THE STATUS OF THE DRIVES ATTACHED TO THE SELECTED MASSBUS SUBSYSTEM TO INHIBIT THIS TYPEOUT, DO NOT RESTART THE PROGRAM FROM ANY OF THE STARTING ADDRESSES, INSTEAD TYPE A 'CONTROL C' ON THE KEYBOARD TO RETURN THE PROGRAM TO COMMAND ENTRY MODE

4.3 PROGRAM ACTION

IN AN EFFORT TO ALLOW CONVERSATION WITH A PROGRAM FOR THE PURPOSE OF CONTROLLING ITS OPERATION AND PARAMETERS THE FOLLOWING CONSTRUCTIONS HAVE BEEN ADOPTED

NOTE1 IN ALL EXAMPLES BRACKETS ARE USED FOR CLARITY AND ARE NOT TYPED BY THE USER

NOTE2 THE CARRRIAGE RETURN TYPED BY THE USER IS INDICATED BY <CR> AND WILL BE ECHOED AS A 'CARRIAGE RETURN-LINE FEED'

< ><CR> PERIOD

A STATEMENT TERMINATOR WHEN TYPED AT THE END OF A LINE (LEGAL ON ALL LINES) IT TELLS THE PARAMETER STRING INTERPRETER (PSI) THIS IS THE END OF CHANGES TO THE CURRENT PARAMETER STRING.

< ><CR> PERIOD PERIOD

THE 'PERIOD PERIOD' TERMINATOR IS TYPED TO INDICATE THE END OF TEST PARAMETER MODIFICATION AND TO SIGNAL

THE START OF TEST EXECUTION

<,><CR> COMMA

THE COMMA IS USED AS A SEPARATOR BETWEEN DRIVE NUMBERS
AND TEST NUMBERS

</> SLASH

A MODIFICATION INDICATOR IF A SLASH FOLLOWS A TEST
NUMBER, THE PROGRAM WILL OPEN THAT TEST FOR PARAMETER
MODIFICATION

< U> CONTROL-U

DELETE THE PRESENT INPUT STRING AND START A NEW
LINE TYPED BY DEPRESSING THE "CONTROL KEY"
(CTRL) AND THEN STRIKING THE "U"

< > RUBOUT

DELETE THE LAST CHARACTER FROM THE INPUT STRING
TYPED BY STRIKING THE "RUBOUT" KEY WHICH WILL
BE ECHOED BY A BACKSLASH ( ) FOLLOWED BY THE
CHARACTER DELETED

4 3 1 CONTROL SWITCH SELECTION

STARTING THE PROGRAM AT ANY OF THE POSSIBLE STARTING ADDRESSES
WITH SW<07>=1 WILL RESULT IN ENTERING THE "CONTROL SWITCH
SETTING" MODE THUS, ALLOWING THE OPERATOR TO SPECIFY THE
DESIRED STATE OF "C SWR"

CONTROL SWITCH SELECTION EXAMPLES

EXAMPLE #1

```
SET SW<07>=0
C SWR=000000 / 400
```

EXAMPLE #2

```
SET SW<07>=0
C SWR=000000 / 220
C SWR=000000 / 220
```

4 3 2 RH11 - RH70 ADDRESS SELECTION

STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC
SELECT OF THE DEFAULT VALUES OF BUS ADDRESS (RPCS1),
VECTOR ADDRESS, AND PRIORITY LEVEL OF THE RH11-RH70
IF THE DEFAULT VAULE OF THE BUS ADDRESS DOES NOT RESPOND
(TIMES OUT) WHEN ADDRESSED, AN ERROR IS REPORTED
AFTER THE ERROR IS REPORTED ONE OF TWO COURSES OF ACTION

WILL BE TAKEN

1 IF THERE IS A MONITOR -- RETURN TO THE MONITOR

2 IF THERE ISN'T A MONITOR -- ASK FOR NEW ADDRESSES

STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR
TO CHANGE THE ADDRESS OF THE RH11 OR RH70 AND THE VECTOR
ADDRESS.

THE PROGRAM ALLOWS THE ADDRESSES TO BE CHANGED ON WHEN THE
PROGRAM IS FIRST STARTED STARTING ADDRESSES 210(8) AND 214(8)
ARE TREATED AS ADDRESSES 200(8) OR 204(8) RESPECTIVELY

ADDRESS SELECTION EXAMPLES

EXAMPLE #1

  RPCS1=176700 / 177200

EXAMPLE #2

  RPCS1=176700 / 176300<CR>
  RHVEC=254 / 260<CR>
  RHPRIO=5 / 6

EXAMPLE #3

  RPCS1=176700<CR>
  RHVEC=254 / 260

EXAMPLE #4

  RH11/RP04 FAILED TO RESPOND TO ADDRESSING
  RPCS1 ERR PC
  176300 XXXXXX
  RPCS1=176300 / 176700

EXAMPLE #5

  RPCS1=176700 / 1776 67 6300<CR>
  RHVEC=254<CR>
  RHPRIO=5<CR>
  RPCS1=176300.

4 3 3 DRIVE AND PARAMETER SELECTION

STARTING THE PROGRAM AT 200 OR 210 WILL RESULT IN AUTOMATIC
SELECTION OF THE DRIVES TO TEST AND THE TESTS TO RUN

STARTING THE PROGRAM AT 204 OR 214 ALLOWS THE OPERATOR
TO SELECT THE DRIVE(S) TO BE TESTED, THE TESTS TO BE EXECUTED,
AND THE PARAMETERS TO USE.

EACH TEST CONTAINS TWO SETS OF CYLINDER LIMIT PARAMETERS PARAMETERS
'LC' AND 'FC' ARE USED BY RP04/5 DRIVES AND PARAMETERS 'LC'' AND

'FC'' ARE USED BY RP06 DRIVES. THE PROGRAM DETERMINES WHICH DRIVE IS BEING TESTED AND SELECTS THE CORRECT SET OF CYLINDER LIMIT VALUES. IF THE PROGRAM IS BEING USED TO TEST A SUBSYSTEM WHICH CONTAINS BOTH RP04/5 AND RP06 DRIVES, THE OPERATOR MUST CHANGE BOTH SETS OF CYLINDER LIMITS IF THE TESTS ARE TO BE MODIFIED FOR ALL DRIVES TESTED.

4 3 3 1 DRIVE AND PARAMETER SELECTION DESCRIPTION

THE FOLLOWING IS A TABLE OF TERMS USED BY THE PSI.

"R"	REPEATS (ITERATIONS)
"FC"	FIRST CYLINDER ADDRESS FOR RP04/5'S
"LC"	LAST CYLINDER ADDRESS FOR RP04/5'S
"FC'"	FIRST CYLINDER ADDRESS FOR RP06'S
"LC'"	LAST CYLINDER ADDRESS FOR RP06'S
"IC"	INCREMENT CYLINDER
"FT"	FIRST TRACK ADDRESS
"LT"	LAST TRACK ADDRESS
"IT"	INCREMENT TRACK
"FS"	FIRST SECTOR ADDRESS
"LS'	LAST SECTOR ADDRESS
"PAT"	PATTERN (USED FOR DATA TEST)
"WDX"	WORD OF PATTERN 0 WHERE X IS 1 TO 16
*"S"	ALL SEEK TESTS (TESTS 0 - 10)
*"T"	ALL TIMING TESTS (TESTS 12 - 15)
*"A"	ALL ADDRESS TESTS (TESTS 16 - 17)
*"D"	THE DATA TEST (TEST 20)
*"E"	THE EXERCISER (TEST 21)

* USED BY THE OPERATOR TO SELECT TEST GROUPS

NOTE ALL NUMBERS WILL BE IN DECIMAL EXCEPT FOR THE PATTERN (PAT) AND WORDS (WDX) SELECTION "PAT" WILL BE SELECTED BY A BIT (I E 001000(8)=PATTERN 9) AND "WDX" WILL BE IN OCTAL

SPECIAL CASES OF CONTROL CHARACTERS

IF < > IS TYPED WHILE A TEST IS OPEN FOR MODIFICATION (</>) AND OTHER TESTS IN THE "TEST COMMAND" STRING ARE TO BE MODIFIED, THE REMAINING TESTS WILL BE UNCHANGED

WHEN THE PROGRAM IS STARTED FROM LOCATION 200 OR 210, TESTS 0-10, 12-20 WILL BE RUN USING ALL AVAILABLE, ONLINE DRIVES IF THE OPERATOR WISHES TO SELECT THE DRIVES TO BE TESTED, THE TESTS TO BE PERFORMED, OR THE PARAMETERS TO BE USED, THE CONVERSATION MODE MAY BE ENTERED BY TYPING A 'CONTROL C' OR BY STARTING THE PROGRAM FROM EITHER LOCATION 204 OR 214

THE PROGRAM WILL THEN RESPOND WITH

DRIVE(S)=

THE FOLLOWING EXAMPLES ASSUME THAT THE OPERATOR IS TO TEST

DRIVE #3 USING TESTS 2 THRU 7 AND TEST 11 AND DOES NOT DESIRE TO CHANGE
THE PARAMETERS (INITIAL CYLINDER ADDRESS, FINAL CYLINDER ADDRESS, ETC.)
THE USER WOULD TYPE '3<CR>' WHICH SAYS 'THIS IS THE END OF DRIVE ENTRY'.
THE PROGRAM WILL THEN REQUEST TEST NUMBERS.

THE TRANSACTION APPEARS AS FOLLOWS:

```
DRIVE(S)=3<CR>
TEST=
```

THE OPERATOR MAY NOW ENTER DESIRED TEST NUMBERS. IN THE EXAMPLE,
HE WANTS TESTS 2 THRU 7 AND TEST 11 SO HE TYPES 2-7<,> (THE 'COMMA'
SEPARATES ENTRIES), 11< ><CR> ('PERIOD' 'CARRIAGE RETURN' -
END OF CHANGES, START TEST EXECUTION)

IT NOW LOOKS LIKE THIS

```
DRIVE(S)=3<CR>
TEST=2-7,11.<CR>
```

IN THE NEXT EXAMPLE, IT IS ASSUMED THAT THE OPERATOR WISHES TO TEST
DRIVE 4 AND TO RUN TESTS 1 AND 3 THRU 11, MODIFYING THE PARAMETERS
FOR TESTS 3 AND 10

THE TRANSACTION WOULD BE AS FOLLOWS

```
DRIVE(S)=4<CR>
TEST=
```

THE OPERATOR NOW ENTERS THE TEST NUMBERS THE TRANSACTION IS
GIVEN BELOW

```
DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
```

NOTICE THIS SAYS SELECT TEST 1, CONTINUE<,>, SELECT TEST 3, OPEN</>,
SELECT TESTS 4-7, CONTINUE<,>, SELECT TEST 10, OPEN</>, SELECT TEST
11, END OF INPUT < >.

THE PROGRAM SCANS THE TEST NUMBER INPUT AND DETERMINES THAT THE
PARAMETERS FOR TEST 3 AND TEST 10 ARE TO BE CHANGED THE OTHER
TESTS WILL NOT BE ALTERED

(THE ENTIRE TRANSACTION IS REPEATED FOR CLARITY)

```
DRIVE(S)=4<CR>
TEST=1,3/4-7,10/11<CR>
  TEST 3
    P=X /                ,WHERE X IS ITERATION
```

THE NEW VALUE FOR 'R' MAY BE ENTERED TERMINATING THE ENTRY WITH
A < > (PERIOD) WILL TERMINATE THE CHANGES FOR THIS TEST, TYPING A
<CR> OR TERMINATING THE ENTRY WITH A <CR> WILL CAUSE THE PROGRAM
TO MOVE TO THE NEXT PARAMETER

```
DRIVE(S)=4<CR>
```

```
        TEST=1,3/4-7,10/11<CR>
          TEST 3
            R=1 / <CR>          ,DO NOT ALTER-BUT CONTINUE
            FC=N /              ,WHERE 'N' IS FIRST CYLINDER ADDRESS
```

IF THE OPERATOR DOES NOT WISH TO CHANGE 'FC', THE FOLLOWING OCCURS·

```
        DRIVE(S)=4<CR>
        TEST=1,3/4-7,10/11 <CR>
          TEST 3
            R=1 / <CR>          ,DO NOT ALTER THIS LINE BUT CONTINUE
            FC=0 / <CR>         ,DO NOT ALTER THIS LINE BUT CONTINUE
            LC=410 /
```

THE PROGRAM RESPONDS WITH THE PREVIOUSLY ASSIGNED PARAMETER FOR LAST CYLINDER ADDRESS IN THIS CASE USING 410 AS THE EXAMPLE. THIS IS WHAT THE OPERATOR INTENDED TO MODIFY AND IS WHY TEST 3 WAS OPENED TO CHANGE THE VALUE TO '20', THE NEW VALUE IS TYPED FOLLOWED BY A 'PERIOD' TERMINATOR (< ><CR>)

THE TOTAL TRANSACTION AND RESPONSE

```
        DRIVE(S)=4<CR>
        TEST=1,3/4-7,10/11<CR>
          TEST 3
            R=1 / <CR>
            FC=0 / <CR>
            LC= 410 / 20 <CR>
          TEST 10
            R=1 /
```

THE PROGRAM HAS LOADED TEST 3 WITH ITS NEW PARAMETERS AND THE PROGRAM IS WAITING FOR CHANGES TO TEST 10'S PARAMETERS

```
        DRIVE(S)=4<CR>
        TEST=1,3/4-7,10/11<CR>
          TEST 3
            R=1 / <CR>
            FC=0 / <CR>
            LC= 410 / 20 <CR>
          TEST 10
            R=1 / 10 <CR>
```

THE OPERATOR TYPES THE NEW VALUE (10) AND TERMINATES THE ENTRY WITH A 'PERIOD' 'CARRIAGE RETURN'

THE PROGRAM NOW LOADS TEST 10 WITH THE NEW PARAMETERS (TEST 11 RETAINS THE PREVIOUSLY ASSIGNED PARAMETERS) AND RESPONDS WITH

```
        DRIVE(S)=
```

SINCE THE USER DID NOT END THE CONVERSATION MODE WITH A 'PERIOD PERIOD', THE PROGRAM HAS LOOPED BACK TO THE BEGINNING LOOKING FOR MORE CHANGES THAT IS TO SAY, AFTER THE ENTRY FOR DRIVE SELECTION, A <,><CR> WILL CAUSE THE TEST MESSAGE TO BE REPEATED AND FURTHER CHANGES CAN BE MADE HOWEVER, AT SOME POINT IN ORDER TO EXECUTE

```
                THE PROGRAM, A 'PERIOD PERIOD' MUST BE TYPED

                IF A SINGLE 'PERIOD' IS TYPED WHILE DRIVE OR TEST NUMBERS ARE BEING
                ENTERED, THE PROGRAM WILL START EXECUTION IMMEDIATELY.  A 'PERIOD
                PERIOD' MUST BE TYPED BEFORE THE PROGRAM WILL EXIT TEST PARAMETER
                CHANGE MODE TO GO TO EXECUTION.

        4 3 3 2 DRIVE AND PARAMETER SELECTION EXAMPLES


                EXAMPLE #1
                ----------

                DRIVE=4 <CR>            ;SELECT DRIVE #4, TERMINATE AND
                                        ,BEGIN EXECUTION USING PREVIOUSLY ASSIGNED
                                        ,PARAMETERS

                EXAMPLE #2
                ----------

                DRIVE=0<CR>             ,SELECT DRIVE #0 AND MAKE CHANGES ","
                TEST=1-5. <CR>          ,RUN TEST 1 THRU 5 ONLY, USE DEFAULT
                                        ,PARAMETERS AND TERMINATE AND EXECUTE."

                EXAMPLE #3
                ----------

                DRIVE=2<CR>             ,SELECT DRIVE #2 AND MAKE CHANGES ","
                TEST=1-5,6/7/10/<CR>    ,RUN TEST 1-5 WITH DEFAULT PARAMETERS, OPEN
                  TEST 6                ;TEST 6,7 AND 10 FOR CHANGES
                    R=1 / <CR>          ;LEAVE 'R' AS IS AND MOVE TO NEXT PARAMETER
                    FC=0 / 10. <CR>     ,SET 'FC' CYLINDER ADDRESS TO 10, END CHANGES
                                        ,TO TEST 6

                  TEST 7
                    R=1 / 50<CR>        ,50 ITERATIONS, MOVE TO NEXT PARAMETER
                    FC=0 / <CR>         ,DO NOT CHANGE 'FC' CYLINDER ADDRESS BUT CONTINUE
                    LC=410 / 50. . <CR> ,TEST 10 IS STILL PENDING AND WILL BE
                                        ;RETAIN ITS PRESENT PARAMETERS


                EXAMPLE #4
                ----------

                DRIVE=0<CR>             .SELECT DRIVE #0 AND MAKE CHANGES
                TEST=S,E <CR>           ,RUN ALL SEEK TESTS AND THE EXERCISER

                EXAMPLE #5
                ----------

                DRIVE=1<CR>
                TEST=S/D<CR>            ,RUN ALL SEEK TESTS (OPEN FOR CHANGES) AND
                  TEST 0                ,THE DATA TEST (WITH DEFAULT PARAMETERS)
                    R=10 / <CR>         ;RUN WITH 10 ITERATIONS
                    FC=0 / 10  <CR>     ,CHANGE FIRST CYLINDER ADDRESS
                                        ,AND START EXECUTION
```

```
                                        ;TESTS 1 - 10 WILL RETAIN THEIR PREVIOUSLY
                                        ;ASSIGNED PARAMETERS.

                    EXAMPLE #6
                    ----------

                    DRIVE=1<CR>
                    TEST=S/<CR>             ;OPEN THE SEEK TESTS (TESTS 0-10)
                      TEST 0
                        R=10 / 100 <CR>     ;CHANGE TO 100 ITERATIONS, TO TO THE NEXT TEST
                      TEST 1
                        R=100 / 1000. <CR>  ;CHANGE 'R' TO 1000 ITERATIONS, MOVE TO NEXT TEST
                      TEST 2
                        R=1 / 10<CR>        ;CHANGE 'R' TO 10 ITERATIONS, GO TO NEXT PARAMETER
                        FC=0 / 50<CR>       ;CHANGE 'FC' TO 50, GO TO NEXT PARAMETER
                        LC=410 / 51. <CR>   ;CHANGE 'LC' TO 51, GO TO THE NEXT TEST
                      TEST 3
                        R=1. <CR>           ;MOVE TO NEXT TEST
                      TEST 4
                        R=1 . <CR>          ;USE TEST 4'S PARAMETERS AND START PROGRAM EXECUTION

                    EXAMPLE #7
                    ----------

                    DRIVE=1<CR>
                    TEST=D/<CR>             ;SELECT AND OPEN THE DATA TEST
                      TEST 20
                        R=1 / 1000<CR>      ;DO 1000 ITERATION OF TEST PATTERN
                        FC=0 / 10<CR>       ;#8 ON CYLINDER 10, TRACK 2, SECTOR 4
                        LC=410 / 10<CR>
                        FC'=0 / <CR>        ;RP06 PARAMETER
                        LC'=814 / <CR>      ;RP06 PARAMETER
                        IC=64 / 0<CR>
                        FT=0 / 2<CR>
                        LT=18 / 2<CR>
                        IT=1 / <CR>
                        FS=0 / 4<CR>
                        LS=22 / 4<CR>
                        PAT=177777 / 400  <CR> ;RUN WITH PATTERN #8

                    EXAMPLE #8
                    ----------

                    DRIVE=1<CR>             ;USE THE SAME PARAMETERS AS IN EXAMPLE
                    TEST=D/<CR>             ;#7, BUT ALSO SPECIFY A DATA PATTERN (PAT #0)
                      TEST 20
                        R=1000 / <CR>
                        FC=10 / <CR>
                        LC=10 / <CR>
                        FC'=0 / <CR>
                        LC'=814 / <CR>
                        IC=0 / <CR>
                        FT=2 / <CR>
                        LT=2 / <CR>
                        IT=1 / <CR>
                        FS=4 / <CR>
```

```
        LS=4 / <CR>
        PAT=000400 / 401<CR> ,RUN WITH PATTERNS #8 & #0 (0=OPERATOR INPUT)
          WD1=165555 / 125252<CR> ;FIRST WORD OF PATTERN 0
          WD2=133333 / 52525. <CR> ,SECOND WORD OF PATTERN 0
                               ,<..> START EXECUTION

    EXAMPLE #9
    ----------

    DRIVE=0,1,4<CR>          ,TEST DRIVES 0,1, AND 4 IN SEQUENCE
    TEST=0-5/<CR>            ,CHANGE TEST 5
      TEST 0
        R=10 / <CR>
        FC=0 / <CR>
        LC=410 / 1<CR>       ,CHANGE LAST CYLINDER FROM 410 TO 1
        FC'=0 / <CR>
        LC'=814 / 2 .<CR>    ,CHANGE THE LAST CYLINDER FOR ALL RP06'S TO
                             ,2   START PROGRAM EXECUTION
```

5. SWITCH SETTINGS

5.1 OPERATIONAL SWITCH SETTINGS

```
WITH SW<15 0>=0 THE PROGRAM WILL PRINT OUT ON
ERRORS AND CONTINUE IN TEST
THE SWITCH SETTINGS ARE.

SW<15>=1.   HALT ON ERROR
SW<14>=1.   LOOP ON TEST
SW<13>=1..  INHIBIT ERROR TYPEOUTS
SW<11>=1. . INHIBIT ITERATIONS
SW<10>=1..  RING BELL ON ERROR
SW<09>=1    LOOP ON ERROR
SW<08>=1.   PRINT ERROR MESSAGE ON LINE PRINTER
SW<07>=1    READ CONTROL SWITCH SETTINGS FROM TTY
SW<06>=1. . INHIBIT TIME REPORTS (TESTS 12-15)
SW<05>=1... REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
SW<04>=1..  INHIBIT WRITES (TEST 20)
SW<03>=1..  INHIBIT WRITE CHECKS (TEST 20)
SW<02>=1... INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
SW<01>=1 .. INHIBIT SOFTWARE COMPARES (TEST 20)
SW<00>=1..  PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I E  AN 11/34)
THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS
NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER   THE
'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8)   THE
SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD
ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'   THE PROGRAM WILL
RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM
IS AT A HIGHER PRIORITY PROCESSING AN RP04/5/6 INTERRUPT   THE
'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE
```

TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED , 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

5 2 CONTROL SWITCH SETTINGS

THE CONTROL SWITCH SETTINGS ARE ENTERED THROUGH THE KEYBOARD

TO ENTER THE CONTROL SWITCH SETTING MODE PLACE SW<07>=1 BEFORE PRESSING START THEN UPON STARTING THE PROGRAM IT WILL TYPE THE PRESENT CONTENTS OF THE CONTROL SWITCH REGISTER (C SWR) AND WAIT FOR THE NEW SETTING TO BE INPUT THE INPUT STRING MUST CONSIST OF 1 TO 6 OCTAL DIGITS, TWO PERIODS ( ), AND A CARRIAGE RETURN.

THE C SWR SETTINGS ARE·

```
C SWR<15>=0   WRITE PACK BEFORE TESTING (TEST16)
         =1   INHIBIT WRITE PACK BEFORE TESTING (TEST16)
C SWR<14>=0 . .NO STALL BETWEEN DRIVE FUNCTIONS
         =1. . .STALL AFTER EVERY DRIVE FUNCTION
C SWR<13>=0 . .USE SPECIFIC STALL TIMES
         =1. . .USE RANDOM STALL TIMES
C SWR<12>=0. . .NO INCREMENTING STALLS IN TEST4
         =1 . .PERFORM INCREMENTING STALLS IN TEST4
C SWR<08>=0.   DO IMPLIED SEEKS WITH DATA TRANSFERS
         =1. . .DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
C SWR<07>=0 . .DO READ HEADER AND  DATA COMMANDS IN TESTS 0-6
         =1 . .DO EXPLICIT SEEK COMMANDS IN TESTS 0-6
C SWR<06>=0. . .60 HZ POWER SOURCE
         =1. . .50 HZ POWER SOURCE
C. SWR<05>=0. . .ALLOW SOFTWARE TIMEOUTS(ENABLE WATCHDOG TIMER)
         =1 . .INHIBIT SOFTWARE TIMEOUTS(DISABLE WATCHDOG TIMER)
C SWR<00>=0.  .OPERATE IN 22 SECTOR (16 BIT) MODE
         =1..  OPERATE IN 20 SECTOR (18 BIT) MODE
```

THE DEFAULT CONDITION OF C SWR<15 00>=0

REFER TO 4 3 1 FOR C SWR SELECTION

6 ERRORS

THERE ARE ANUMBER OF ERRORS THAT CAN OCCUR IN THIS PROGRAM.
WHEN AN ERROR IS ENCOUNTERED, THE CALL TO THE ERROR ROUTINE
IS MADE AND IF SW<13> IS NOT SET, AN ERROR MESSAGE PERTAINING
TO THE ERROR WILL BE TYPED EACH ERROR TYPEOUT WILL CONTAIN
THE FOLLOWING·

1. AN ERROR MESSAGE
2. A DATA HEADER
3. A DATA STRING

REFER TO THE FOLLOWING SECTION FOR THE DIFFERENT ERRORS
THAT CAN OCCUR

6.1 ERROR TYPES

THE ERRORS THAT OCCUR IN THIS PROGRAM FALL INTO THREE
(3) CATEGORIES DEFINED AND EXPLAINED AS FOLLOWS

6.1.1 DRIVER ERROR

THESE ERRORS WILL BE DETECTED BY THE RH11/RP04/5/6 DRIVER
THERE ARE TWO CLASSES OF DRIVER ERRORS; THOSE THAT
CAN NOT BE IDENTIFIED IN A MANNER THAT ALLOWS THE
INFORMATION TO BE RETURNED TO A "DATA PARAMETER BLOCK"
(DPB) AND THOSE THAT CAN. THE FIRST CLASS WILL BE
REPORTED BY ERROR CALLS (EMT'S) 1-5 WITHIN THE DRIVER
THE SECOND CLASS WILL PASS THE ERROR CODES TO THE
STATUS/ERROR WORD (DPB+16) OF THE PROPER DPB

6.1.2 NON-FATAL ERRORS

THESE ERRORS WILL BE DUE TO "DISK" OR "DATA" FAILURES
WHICH WILL BE REPORTED AS THEY OCCUR AFTER REPORTING
THE ERROR THE PROGRAM WILL CONTINUE TESTING

6.1.3 FATAL ERRORS

THIS TYPE OF ERROR WILL BE THE RESULT OF ANY KIND
OF ERROR THAT INHIBITS THE PROGRAM FROM TESTING THE DISK

THIS ERROR WILL BE REPORTED WHEN IT OCCURS, THEN THE PROGRAM
WILL ABORT THE TEST AND GO TO THE END OF PROGRAM

6 2 ERROR RECOVERY

6.2.1 PRETEST ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED THEN DEPENDING
ON HOW THE PROGRAM WAS STARTED IT WILL ASK FOR THE DRIVES AND
ADDRESSES FOR TESTING OR RETURN TO MONITOR

6 2 2 NON-FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED AND
THE PROGRAM WILL CONTINUE IN TEST

6 2 3 FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THE PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

7 RESTRICTIONS

THE PROGRAM WILL TEST THE DRIVES IN EITHER 16 BIT MODE OR IN 18 BIT MODE DEPENDING ON THE SETTING OF 'S.SWR<00>'. IF 'C.SWR<00>' IS 0, ALL OF THE DRIVES WILL BE TESTED IN 16 BIT MODE, IF 'C.SWR<00> IS 1, ALL OF THE DRIVES WILL BE TESTED IN 18 BIT MODE THE PROGRAM HAS NO PROVISIONS FOR TESTING DRIVES WITH INTERMIXED PACKS OR TESTING BOTH 16 BIT MODE AND 18 BIT MODE DRIVES ON THE SAME SYSTEM. ACT11 AUTOMATIC MODE ASSUMES 16 BIT MODE.

BEFORE THE PROGRAM IS STARTED, PROPERLY FORMATTED PACKS MUST BE MOUNTED ON THE DRIVES WHICH WILL BE TESTED THE PROGRAM ASSUMES A PROPERLY FORMATTED PACK. THE FORMAT OF THE PACK IS NOT ALTERED BY THE PROGRAM

8 MISCELLANEOUS

8 1 EXECUTION TIME

THE PROGRAM REQUIRES APPROXIMATELY 15 MINUTES TO MAKE ONE PASS WITH RP04/5 DRIVES AND APPROXIMATELY 16 5 MINUTES TO A PASS WITH RP06 DRIVES THIS ASSUMES THE DEFAULT TEST SEQUENCE (TESTS 0-10, 12-20) AND DEFAULT TEST PARAMETERS

8 2 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 1100

8 3 TIMING TESTS (TESTS 12-15) PRINTOUTS

AT THE COMPLETION OF EACH OF THE TIMING TESTS THE TIME OF THE MINIMUM SEEK, MAXIMUM SEEK, AND THE AVERAGE OF ALL OF THE SEEKS PERFORMED ARE TYPED ON THE TTY THE NUMBER OF SEEKS THAT HAD TIMES BELOW THE MINIMUM TIME ALLOWED WILL BE TYPED ON THE SAME LINE AS THE MINIMUM TIME THE NUMBER ABOVE THE MAXIMUM WILL BE TYPED ON THE SAME LINE AS THE MAXIMUM TIME, AND THE TOTAL NUMBER OF SEEKS PERFORMED WILL BE ON THE SAME LINE AS THE AVERAGE

NOTE THE PROGRAM STALLS FOR 2 MILLISECONDS BETWEEN SEEK ORDERS, THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES THE 2 MILLISECOND STALL BETWEEN SEEK ORDERS IS SPECIFIED BY

THE RP04 VENDOR. THE SEEK TIMES SPECIFIED FOR THE RP04
ARE POSITIONER MOVEMENT TIMES ONLY AND ARE NOT A MEASUREMENT
OF EFFECTIVE SEEK TIME.

8 3 1 TIMING TOLERANCES

1 TEST 12 -- ROTATIONAL SPEED TIMES

60 HZ
MINIMUM=16340 US
MAXIMUM=17000 US
NOMINAL=16670 US

50 HZ
MINIMUM=16250 US
MAXIMUM=17090 US
NOMINAL=16670 US

2 TEST 13 -- ONE CYLINDER SEEK TIMES

MAXIMUM=10000 US
NOMINAL=7000 US

3 TEST 14 -- ACCESS TIME MEASUREMENT

MAXIMUM=30000 US
NOMINAL=28000 US

4 TEST 15 -- MAXIMUM SEEK TIMES

MAXIMUM=52000 US
NOMINAL=50000 US

8 3 2 TIMING TESTS PRINTOUT EXAMPLES

EXAMPLE #1

ROTATIONAL SPEED TIMES
MIN=16670 US
MAX=16690 US
AVG=16680 US 10 SEEKS TIMED

ONE CYLINDER SEEK TIMES
 * FORWARD
MIN=5350 US
MAX=6920 US
AVG=5550 US 409 SEEKS TIMED
 * REVERSE
MIN=5140 US
MAX=5960 US
AVG=5430 US 410 SEEKS TIMED

ACCESS TIME MEASUREMENTS
 * FORWARD

```
                    MIN=27770 US
                    MAX=28640 US
                    AVG=28230 US  128 SEEKS TIMED
                     * REVERSE
                    MIN=27990 US
                    MAX=28550 US
                    AVG=28220 US  128 SEEKS TIMED

                    MAXIMUM SEEK TIMES
                     * FORWARD
                    MIN=49990 US
                    MAX=51980 US
                    AVG=51010 US  128 SEEKS TIMED
                     * REVERSE
                    MIN=48120 US
                    MAX=50650 US
                    AVG=49340 US  128 SEEKS TIMED


                    EXAMPLE #2
                    ----------

                    ROTATIONAL SPEED TIMES
                    MIN=16670 US
                    MAX=16690 US
                    AVG=16680 US  10 SEEKS TIMED

                    ONE CYLINDER SEEK TIMES
                     * FORWARD
                    MIN=5470 US
                    MAX=10940 US  3 ABOVE THE MAXIMUM OF 10000 US
                    AVG=5830 US  409 SEEKS TIMED
                     * REVERSE
                    MIN=5040 US
                    MAX=5970 US
                    AVG=5330 US 410 SEEKS TIMED

                    ACCESS TIME MEASURMENTS
                     * FORWARD
                    MIN=29730 US
                    MAX=31620 US  73 ABOVE THE MAXIMUM OF 30000 US
                    AVG=30320 US  128 SEEKS TIMED
                     * REVERSE
                    MIN=28620
                    MAX=31230 US  128 ABOVE THE MAXIMUM OF 30000 US
                    AVG=30800 US  128 SEEKS TIMED

                    MAXIMUM SEEK TIMES
                     * FORWARD
                    MIN=53510 US
                    MAX=54240 US  128 ABOVE THE MAXIMUM OF 52000 US
                    AVG=54020 US  128 SEEKS TIMED
                     * REVERSE
                    MIN=52050 US
                    MAX=54550 US  128 ABOVE THE MAXIMUM OF 52000 US
                    AVG=52210 US  128 SEEKS TIMED
```

8 4 END OF TEST

WITH ALL SWITCHES ON A "0" AN "END OF PASS" MESSAGE WILL BE TYPED AT THE COMPLETION OF TESTING A DRIVE AND THE "END OF TEST" TYPEOUT WILL OCCUR WHEN ALL DRIVES HAVE BEEN TESTED

9 PROGRAM DESCRIPTION

THIS PROGRAM CONTAINS NINTEEN TESTS NUMBERED 0-22 IN OCTAL TESTS 0-7 & 11 WILL READ THE CYLINDER, TRACK, AND SECTOR INFORMATION FROM THE HEADER, USING A "READ HEADER AND DATA" COMMAND, AND THEN CHECK THE INFORMATION FOR VALIDITY. THUS, INSURING THE SEEK OPERATION FUNCTIONS PROPERLY. TESTS 12-15 WILL MEASURE THE ROTATIONAL SPEED, THE ONE CYLINDER SEEK, THE ACCESS TIME, AND THE MAXIMUM SEEK TIMES TO ENSURE THEY ARE ALL WITHIN THE TOLERANCES ALLOWED. TEST 16 AND 17 ENSURES THE SECTOR AND TRACK ADDRESSING CIRCUITRY WORKS PROPERLY. TEST 20 VERIFIES THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONAL AND TEST 21 WILL STRESS AND CHECK THE READ/WRITE AND SERVO SYSTEMS.

THE PROGRAM WILL START BY IDENTIFING ITSELF AND DETERMINING ALL DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH THE LOWEST NUMBERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER ALL OF THE DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE (TESTS 0-10,12-20) WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE DRIVE TO BE TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS, AN "END OF PASS" MESSAGE WILL BE TYPED AT THE COMPLETION OF EACH PASS, AND AN "END OF TEST" MESSAGE WILL BE TYPED AFTER TESTING ALL DRIVES

REFER TO THE FOLLOWING SECTIONS FOR DETAILED DESCRIPTIONS OF EACH TEST

9.1 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

THE PARAMETEPS USED BY THE TEST ARE GIVEN BELOW

R	-	200
LC	-	410
LC'	-	814
FT	-	0
FS	-	0

9 2 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.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW

R	-	100
FC	-	0
LC	-	128
FC'	-	0
LC'	-	256
IC	-	0
FT	-	0
LT	-	0
FS	-	0
LS	-	0

9 3 TEST 2 - INCREMENTAL SEEK TEST

THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC" WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS "LC' REVERSE SEEK CYCLES ARE INITIATED, STARTING AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC" UNTIL "NC" IS LESS THAN "FC" AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	1
FT	-	0
FS	-	0

9 4 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 VERIFY PROPER OPERATION

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW

R	-	8
FC	-	0
LC	-	256
FC'	-	0

LC'	-	256
IC	-	1
FT	-	0
FS	-	0

9 5 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.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST.

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	1
FT	-	0
FS	-	0

9 6 TEST 5 - CONVERGING/DIVERGING SEEK TEST

THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
LESS THAN THE INITIAL VALUE OF "NC1" AT THE COMPLETION OF
EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
ENSURE PROPER OPERATION "NC1" AND "NC2" DEFAULT TO
'FC" AND "LC" RESPECTIVELY
THE FOLLOWING PARAMETERS ARE USED BY THE TEST

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	1
FT	-	0
FS	-	0

9 7 TEST 6 - SERVO ADDRESSING LOGIC NOISE GENERATOR TEST

IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5 NOW "NC" IS UPDATED
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

THE FOLLOWING PARAMETERS ARE USED BY THE TEST

```
                        R       -       1
                        FC      -       0
                        LC      -     410
                        FC'     -       0
                        LC'     -     814
                        IC      -       1
                        FT      -       0
                        FS      -       0

9 8     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'.

        THE FOLLOWING PARAMETERS ARE USED BY THE TEST

                        R       -     5000
                        FC      -       0
                        LC      -     410
                        FC'     -       0
                        LC'     -     814
                        FT      -       0
                        LT      -      18

9 9     TEST 10 - SERVO SETTLE DOWN TEST

        THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT
        THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE
        RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+IC'
        ('NC1' STARTS AT VALUE 'FC').  AT THE COMPLETION OF 1000 (10) SEEKS,
        'NC1' IS INCREMENTED BY VALUE 'IC' AND THE SEQUENCE IS REPEATED
        THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'

        WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD
        REGISTER (RPLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING  INTO
        POSITION.  THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
        FOR THAT SECTOR.  IF THE DRIVE'S POSITIONER HAS NOT SETTLED DOWN OR
        IF THE POSITIONER IS NOT ON CYLINDER (IF THE DRIVE IS AN RP04, THE
        OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 US), THE DRIVE
        WILL REPORT A 'WRU' ERROR   (RP05/6'S MAY ALSO REPORT 'NHS' ERROR UNDER
        ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED
        CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PCAK WITH
        MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE

        THIS TEST USES THE EXTENTION BITS IN THE LOOK-AHEAD REGISTER TO DETERMINE
        WHETHER OR NOT IT CAN PICK UP THE SECTOR ROTATING INTO POSITION   THE
        TEST IS OPTIMIZED SUCH THAT IF THE DRIVE SIGNALS SEEK DONE WITHIN
        THE FIRST 80% OF THE SECTOR CURRENTLY UNDER THE HEAD, THE TEST WILL
        TRY TO ADDRESS THE NEXT SECTOR   BASED ON OBSERVATION, THE PROGRAM
        IS ABLE TO START THE OPERATION WITHOUT LOSING A REVOLUTION MOST OF
        THE TIME
```

THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS THE NECESSARY TIME
DEPENDENT PARAMETERS OCCUR WITHIN THE REQUIRED TIME RANGE FREQUENTLY
ENOUGH TO PERMIT THIS TEST TO BE EFFECTIVE

THE FOLLOWING PARAMETERS ARE USED BY THE TEST

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	100
FT	-	0

9 10 TEST 11 - ALL SEEKS TEST

THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER
TO ALL OTHER CYLINDERS.

BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER
BETWEEN 'FC' AND 'LC' FROM CYINDER 'FC'. THE BEGINNING CYLINDER
ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER
ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC' THE SEQUENCE
CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED

THE FOLLOWING PARAMETERS ARE USED BY THIS TEST

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	1
FT	-	0
FS	-	0

9 11 TEST 12 - ROTATIONAL SPEED TIMING TEST

THIS TEST WILL START A SEARCH TO CYLINDER 0, TRACK 0, SECTOR
0 AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS SET AGAIN
AND THE OPERATION IS TIMED THIS PROCEDURE IS REPEATED 10
TIMES THEN THE AVERAGE TIME IS CALCULATED AND CHECKED TO
ENSURE IT IS WITHIN TOLERANCE

16 67 MS/REV + OR - 2% IF 60HZ
16 67 MS/REV + OR - 2 5% IF 50HZ

THE FOLLOWING PARAMETERS ARE USED BY THE TEST

R	-	1
FC	-	0
FC'	-	0
FT	-	0
FS	-	0

9 12 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.

THE TEST USES THE FOLLOWING PARAMETERS.

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814

9 13 TEST 14 - ACCESS TIME MEASUREMENT

THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO CYLINDER 'LC', THEN A REVERSEK FROM CYLINDER 'LC' TO CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASURMENT. THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256 SEEKS). THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS CYLINDER 'LC' DEFAULTS TO 136 (10) FOR AN RP04/5 OR TO 255 (10) FOR AN RP06.

THE TEST USES THE FOLLOWING PARAMETERS

R	-	1
FC	-	0
LC	-	136
FC'	-	0
LC'	-	255

9 14 TEST 15 - MAXIMUM SEEK TIMING TEST

THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK TIME THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256 SEEKS). THE MAXIMUM SEEK TIME MUST BE LESS THAN 54 MS 'LC' DEFAULTS TO 410 (10) FOR RP04/5'S AND TO 814 (10) FOR RP06'S

THE TEST USES THE FOLLOWING PARAMETERS

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814

9 15 TEST 16 - SECTOR ADDRESSING TEST

THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT". THE
DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR
BEING WRITTEN. A WRITE CHECK IS PERFORMED, THE BUFFER IS
CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED. THEN SECTOR 0
IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED. THEN
SECTOR 1 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED.
THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
REWRITE SECTOR 21 AND WRITE CHECK SECTORS 0-21.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
FC'	-	0
FT	-	0

9 16 TEST 17 - TRACK ADDRESSING TEST

THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
GETTING ITS OWN TRACK ADDRESS.
A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO INSURE
THE DATA IS VALID. THEN TRACK 0 IS REWRITTEN AND TRACK 1
THROUGH TRACK 18 IS WRITE CHECKED. THEN TRACK 1 IS
REWRITTEN AND TRACK 2 THROUGH TRACK 18 IS WRITE CHECKED
THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING TRACK 17
AND WRITE CHECKING TRACK 18

THE TEST USES THE FOLLOWING PARAMETERS

R	-	1
FC	-	0
FC'	-	0
FS	-	0

9 17 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
133333	004000	173777	155555	177757	066667	177777	000000
165555	010000	167777	172666	177767	153333	177777	000000
133333	020000	157777	155555	177773	066667	177777	000000
165555	040000	137777	172666	177775	153333	177777	000000
133333	100000	077777	155555	177776	066667	177777	000000

THE TEST USES THE FOLLOWING PARAMETERS

R	-	1
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814
IC	-	64
FT	-	0
LT	-	18
IT	-	1
FS	-	1
LS	-	0
PAT	-	177777

9 18 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 ADDRRESS
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

THE TEST USES THE FOLLOWING PARAMETERS

R	-	20000
FC	-	0
LC	-	410
FC'	-	0
LC'	-	814

9 19 TEST 22 - RP04 ACCESS TIME ADJUSTMENT TEST

THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 136 TO ALLOW THE
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

THE TEST USES THE FOLLOWING PARAMETERS

R	-	5000
FC	-	0
LC	-	136
FC'	-	0
LC'	-	255

10 PROGRAM LISTING

```
1426                            TITLE  MD-11-DZRJA-B RP04/5/6 MECHANICAL AND READ/WRITE TEST
1427                           ;*COPYRIGHT (C) 1976,1977
1428                           ,*DIGITAL EQUIPMENT CORP.
1429                           ,*MAYNARD, MASS. 01754
1430                           ;*
1431                           ;*PROGRAM BY C. HESS
1432                           ,*
1433                           ,*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1434                           ,*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977
1435                           ,*
1436
1437                            SBTTL  CONTROL SWITCH SETTINGS
1438                           ,*
1439                           ,*    SWITCH  STATE              USE
1440                           ,*    ------  -----    ------------------------------------
1441                           ;*      15      0      WRITE PACK BEFORE TESTING (TEST 21)
1442                           ,*              1      INHIBIT WRITING PACK BEFORE TESTING (TEST 21)
1443                           ,*      14      0      NO STALL BETWEEN DRIVE FUNCTIONS
1444                           ,*              1      STALL AFTER EVERY DRIVE FUNCTION
1445                           ,*      13      0      USE SPECIFIC STALL TIME
1446                           ,*              1      USE RANDOM STALL
1447                           ,*      12      0      NO INCREMENTING STALL IN TEST 4
1448                           ,*              1      DO INCREMENTING STALL IN TEST 4
1449                           ,*       8      0      DO IMPLIED SEEKS WITH DATA TRANSFERS
1450                           ,*              1      DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
1451                           ,*       7      0      DO "READ HEADER AND DATA" IN TESTS 0-11
1452                           ;*              1      DO EXPLICIT SEEKS IN TESTS 0-11
1453                           ,*       6      0      60 HZ
1454                           ,*              1      50 HZ
1455                           ,*       5      0      RUN WATCHDOG TIMER
1456                           ,*              1      INHIBIT WATCHDOG TIMER
1457                           ,*       0      0      TEST DRIVE(S) IN 22 SECTOR (16 BIT) MODE
1458                           ,*              1      TEST DRIVE(S) IN 20 SECTOR (18 BIT) MODE
1459
1460                            SBTTL  OPERATIONAL SWITCH SETTINGS
1461                           ;*
1462                           ;*      SWITCH                 USE
1463                           ,*      ------          --------------------
1464                           ,*        15            HALT ON ERROR
1465                           ,*        14            LOOP ON TEST
1466                           ,*        13            INHIBIT ERROR TYPEOUTS
1467                           ,*        10            BELL ON ERROR
1468                           ;*         9            LOOP ON ERROR
1469                           ,*         8            PRINT ERROR MESSAGE ON LINE PRINTER
1470                           ,*         7            READ CONTROL SWITCH SETTINGS FROM TTY
1471                           ,*         6            INHIBIT TIME REPORTS (TESTS 12-15)
1472                           ,*         5            REPORT ONE ERROP PER SECTOR (TESTS 16 & 17)
1473                           ,*         4            INHIBIT WRITES (TEST 15)
1474                           ,*         3            INHIBIT WRITE CHECKS (TEST 20)
1475                           ,*         2            INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
1476                           ,*         1            INHIBIT SOFTWARE COMPARES (TEST 20)
1477                           ,*         0            PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)
1478
1479                            SBTTL  TRAP CATCHER
```

```
1480
1481            000000                          .=0
1482                                            ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1483                                            ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1484                                            ,*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1485            000174                          .=174
1486  000174    000000                        DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
1487  000176    000000                        SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
1488
1489                                            .SBTTL  ACT11 HOOKS
1490
1491                                            ,;**************************************************************
1492                                            ;HOOKS REQUIRED BY ACT11
1493            000200                                  $SVPC=.                 ;SAVE PC
1494            000046                                  .=46
1495  000046    017614                                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN  $EOP
1496            000052                                   =52
1497  000052    000000                                  .WORD   0               ,,2)SET LOC 52 TO ZERO
1498            000200                                  .=$SVPC                 ;; RESTORE PC
1499
1500                                             SBTTL  STARTING ADDRESSES
1501            000200                           =200
1502                                            ,*200 = NORMAL START
1503  000200    000137  004636                          JMP     @#START1
1504                                            ,*204 = SELECT OPERATING PARAMETERS
1505  000204    000137  004660                          JMP     @#START2
1506                                            ,*210 = SELECT RH11/RP04/5/6 ADDRESSES
1507  000210    000137  004626                          JMP     @#START3
1508                                            ,*214 = COMBINATION OF 204 AND 210
1509  000214    000137  004650                          JMP     @#START4
1510
1511                                             SBTTL  BASIC DEFINITIONS
1512
1513                                            ,*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1514            001100                          STACK=  1100
1515                                            .EQUIV  EMT,ERROR       ,,BASIC DEFINITION OF ERROR CALL
1516                                             EQUIV  IOT,SCOPE       ,,BASIC DEFINITION OF SCOPE CALL
1517
1518                                            ,*MISCELLANEOUS DEFINITIONS
1519            000011                          HT=     11              ,,CODE FOR HORIZONTAL TAB
1520            000012                          LF=     12              ,,CODE FOR LINE FEED
1521            000015                          CR=     15              ,,CODE FOR CARRIAGE RETURN
1522            000200                          CRLF=   200             ,,CODE FOR CARRIAGE RETURN-LINE FEED
1523            177776                          PS=     177776          ,,PROCESSOR STATUS WORD
1524                                             EQUIV  PS,PSW
1525            177774                          STKLMT= 177774          ,,STACK LIMIT REGISTER
1526            177772                          PIRQ=   177772          ;,PROGRAM INTERRUPT REQUEST REGISTER
1527            177570                          DSWR=   177570          ,,HARDWARE SWITCH REGISTER
1528            177570                          DDISP=  177570          ,,HARDWARE DISPLAY REGISTER
1529
1530                                            ,*GENERAL PURPOSE REGISTER DEFINITIONS
1531            000000                          R0=     %0              ,,GENERAL REGISTER
1532            000001                          R1=     %1              ,;GENERAL REGISTER
1533            000002                          R2=     %2              ,,GENERAL REGISTER
1534            000003                          R3=     %3              ,,GENERAL REGISTER
1535            000004                          R4=     %4              ,,GENERAL REGISTER
```

```
1536            000005          R5=     %5              ;;GENERAL REGISTER
1537            000006          R6=     %6              ;;GENERAL REGISTER
1538            000007          R7=     %7              ;;GENERAL REGISTER
1539            000006          SP=     %6              ;;STACK POINTER
1540            000007          PC=     %7              ;;PROGRAM COUNTER
1541
1542                            ;*PRIORITY LEVEL DEFINITIONS
1543            000000          PR0=    0               ;;PRIORITY LEVEL 0
1544            000040          PR1=    40              ;;PRIORITY LEVEL 1
1545            000100          PR2=    100             ;;PRIORITY LEVEL 2
1546            000140          PR3=    140             ;;PRIORITY LEVEL 3
1547            000200          PR4=    200             ;;PRIORITY LEVEL 4
1548            000240          PR5=    240             ;;PRIORITY LEVEL 5
1549            000300          PR6=    300             ;;PRIORITY LEVEL 6
1550            000340          PR7=    340             ;;PRIORITY LEVEL 7
1551
1552                            ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1553            100000          SW15=   100000
1554            040000          SW14=   40000
1555            020000          SW13=   20000
1556            010000          SW12=   10000
1557            004000          SW11=   4000
1558            002000          SW10=   2000
1559            001000          SW09=   1000
1560            000400          SW08=   400
1561            000200          SW07=   200
1562            000100          SW06=   100
1563            000040          SW05=   40
1564            000020          SW04=   20
1565            000010          SW03=   10
1566            000004          SW02=   4
1567            000002          SW01=   2
1568            000001          SW00=   1
1569                             EQUIV  SW09,SW9
1570                             EQUIV  SW08,SW8
1571                             EQUIV  SW07,SW7
1572                             EQUIV  SW06,SW6
1573                             EQUIV  SW05,SW5
1574                             EQUIV  SW04,SW4
1575                             EQUIV  SW03,SW3
1576                             EQUIV  SW02,SW2
1577                             EQUIV  SW01,SW1
1578                             EQUIV  SW00,SW0
1579
1580                            ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1581            100000          BIT15=  100000
1582            040000          BIT14=  40000
1583            020000          BIT13=  20000
1584            010000          BIT12=  10000
1585            004000          BIT11=  4000
1586            002000          BIT10=  2000
1587            001000          BIT09=  1000
1588            000400          BIT08=  400
1589            000200          BIT07=  200
1590            000100          BIT06=  100
1591            000040          BIT05=  40
```

```
1592    000020          BIT04=  20
1593    000010          BIT03=  10
1594    000004          BIT02=  4
1595    000002          BIT01=  2
1596    000001          BIT00=  1
1597                    .EQUIV  BIT09,BIT9
1598                    .EQUIV  BIT08,BIT8
1599                    .EQUIV  BIT07,BIT7
1600                     EQUIV  BIT06,BIT6
1601                    .EQUIV  BIT05,BIT5
1602                     EQUIV  BIT04,BIT4
1603                     EQUIV  BIT03,BIT3
1604                     EQUIV  BIT02,BIT2
1605                     EQUIV  BIT01,BIT1
1606                    .EQUIV  BIT00,BIT0
1607
1608                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1609    000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
1610    000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1611    000014          TBITVEC=14              ;;"T" BIT
1612    000014          TRTVEC= 14              ;;TRACE TRAP
1613    000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
1614    000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1615    000024          PWRVEC= 24              ;;POWER FAIL
1616    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1617    000034          TRAPVEC=34              ;;"TRAP" TRAP
1618    000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
1619    000064          TPVEC=  64              ;;TTY PRINTER VECTOR
1620    000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1621                    ;;*****************************************************************
1622
1623                     SBTTL  RH11 REGISTERS
1624
1625                    ;;*****************************************************************
1626
1627                    ;CONTROL AND STATUS REGISTER 1 (RPCS1)
1628
1629    000100          IE=     100             ;INTERRUPT ENABLE (BIT #6)
1630    000200          RDY=    200             ;READY (BIT #7)
1631    000400          A16=    400             ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
1632    001000          A17=    1000            ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
1633    002000          PSEL=   2000            ;PORT SELECT (BIT #10)
1634    020000          MCPE=   20000           ;MASSBUSS PARITY ERROR (BIT #13)
1635    040000          TRE=    40000           ;TRANSFER ERROR (BIT #14)
1636                    ;SC=    100000          ;SPECIAL CONDITION (BIT #15)
1637
1638                    ;WORD COUNT REGISTER (RPWC)
1639                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1640
1641                    ;BUS ADDRESS REGISTER (RPBA)
1642                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1643
1644                    ;CONTROL AND STATUS REGISTER 2 (RPCS2)
1645
1646    000001          US1=    1               ;UNIT SELECT (BIT #0)
1647    000002          US2=    2               ;UNIT SELECT (BIT #1)
```

```
                                ;UNIT SELECT (BIT #2)
                                ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
                                ;MASSBUS PARITY TEST (BIT #4)
                                ;CLEAR (BIT #5)
                                ;INPUT READY (BIT #6)
                                ;OUTPUT READY (BIT #7)
                                ;MASS BUS PARITY ERROR (BIT #8)
                                ;MISSED TRANSFER ERROR (BIT #9)
                                ;PROGRAM ERROR (BIT #10)
                                ;NON EXISTENT MEMORY (BIT #11)
                                ;NON EXISTENT DRIVE (BIT #12)
                                ;UNIBUS PARITY ERROR (BIT #13)
                                ;WRITE CHECK ERROR (BIT #14)
                                ,DATA LATE (BIT #15)

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


                                ,,*************************************************************

                                 SBTTL  RP04/5/6 REGISTERS

                                ,,*************************************************************

                                ,CONTROL AND STATUS 1 REGISTER  (#00)

                                ,GO BIT (BIT #0)
                                ,FUNCTION CODE BIT #1
                                ,FUNCTION CODE BIT #2
                                ,FUNCTION CODE BIT #3
                                ,FUNCTION CODE BIT #4
                                ,FUNCTION CODE BIT #5
                                ,DEVICE AVAILABLE (BIT #11)

                                ,DRIVE STATUS REGISTER (RPDS1) (#01)

                                DRIVE FORWARD 5"/SEC   (BIT #0)
                                ,DRIVE FORWARD 20"/SEC   (BIT #1)
                                ,DRIVE TO INNER GUARD BAND (BIT #2)
                                ,GO REVERSE (BIT #3)
                                ,DIFFERENCE LESS THAN 64 (BIT #4)
                                ,DIFFERENCE EQUALS 1 (BIT #5)
                                ,VOLUME VALID (BIT #6)
                                ,DRIVE READY (BIT #7)
                                ,DRIVE PRESENT (BIT #8)
                                ,PROGRAMABLE (BIT #9)
                                ,LAST SECTOR TRANSFERRED (BIT #10)
                                ,WRITE LOCK (BIT #11)
                                ,MEDIUM ON-LINE (BIT #12)
                                ,POSITIONING OPERATION IN PROGRESS (BIT #13)
                                ,COMPOSITE ERROR (BIT #14)
                                ,ATTENTION ACTIVE (BIT #15)

                                ,ERROR REGISTER #01 (RPER1) (#02)
```

```
1704    000001          ILF=    1               ,ILLEGAL FUNCTION (BIT #0)
1705    000002          ILR=    2               ;ILLEGAL REGISTER (BIT #1)
1706    000004          RMR=    4               ;REGISTER MODIFICATION REFUSED (BIT #2)
1707    000010          PAR=    10              ;PARITY ERROR (BIT #3)
1708    000020          FER=    20              ;FORMAT ERROR (BIT #4)
1709    000040          WCF=    40              ;WRITE CLOCK FAIL (BIT #5)
1710    000100          ECH=    100             ;ECC HARD ERROR (BIT #6)
1711    000200          HCE=    200             ;HEADER COMPARE ERROR (BIT #7)
1712    000400          HCRC=   400             ;HEADER CRC ERROR (BIT #8)
1713    001000          AOE=    1000            ;ADDRESS OVERFLOW ERROR (BIT #9)
1714    002000          IAE=    2000            ;INVALID ADDRESS ERROR (BIT #10)
1715    004000          WLE=    4000            ,WRITE LOCK ERROR (BIT #11)
1716    010000          DTE=    10000           ,DRIVE TIMING ERROR (BIT #12)
1717    020000          OPI=    20000           ;OPERATION INCOMPLETE (BIT #13)
1718    040000          UNS=    40000           ,DRIVE UNSAFE (BIT #14)
1719    100000          DCK=    100000          ,DATA CHECK ERROR (BIT 15)
1720
1721                    ,MAINTAINABILITY REGISTER (RPMR)(#03)
1722
1723    000001          DMD=    1               ,DIAGINOSTIC MODE (BIT #0)
1724    000002          MCLK=   2               ;MAINTAINABILITY CLOCK (BIT #1)
1725    000004          MINX=   4               ,MAINTAINABILITY INDEX (BIT #2)
1726    000010          MSTCK=  10              ,MAINTAINABILITY SECTOR CLOCK (BIT #3)
1727    000020          MRD=    20              ,MAINTAINABILITY READ (BIT #4)
1728    000040          MWR=    40              ,MAINTAINABILITY WRITE (BIT #5)
1729    000200          DTSY=   200             ,MAINTAINABILITY SYNC DETECTED (BIT #7)
1730
1731                    ,ATTENTION SUMMARY PSEUDO-REGISTER (RPAS) (#04)
1732
1733    000001          AT0=    1               ,DEVICE 0 (BIT #0)
1734    000002          AT1=    2               ,DEVICE 1 (BIT #1)
1735    000004          AT2=    4               ,DEVICE 2 (BIT #2)
1736    000010          AT3=    10              ,DEVICE 3 (BIT #3)
1737    000020          AT4=    20              ,DEVICE 4 (BIT #4)
1738    000040          AT5=    40              ,DEVICE 5 (BIT #5)
1739    000100          AT6=    100             ,DEVICE 6 (BIT #6)
1740    000200          AT7=    200             ,DEVICE 7 (BIT #7)
1741
1742                    ,DESIRED SECTOR/TRACK ADDRESS REGISTER (RPDA) (#05)
1743                    ,(EACH BIT IS CALLED BY BIT NUMBER)
1744
1745                    ,DRIVE TYPE REGISTER (RPDT) (#06)
1746
1747    000001          DT00=   1               ,DRIVE TYPE NUMBER BIT 1
1748    000002          DT01=   2               ,DRIVE TYPE NUMBER BIT 2
1749    000004          DT02=   4               ,DPIVE TYPE NUMBER BIT 3
1750    000010          DT03=   10              ,DRIVE TYPE NUMBER BIT 4
1751    000020          DT04=   20              ,DRIVE TYPE NUMBER BIT 5
1752    000040          DT05=   40              ,DRIVE TYPE NUMBER BIT 6
1753    000100          DT06=   100             ,DRIVE TYPE NUMBER BIT 7
1754    000200          DT07=   200             ,DRIVE TYPE NUMBER BIT 8
1755    000400          DT08=   400             ,DRIVE TYPE NUMBER BIT 9
1756    004000          DRQ=    4000            ,DRIVE REQUEST REQUIRED (BIT #11)
1757    020000          MOH=    20000           ,MOVING HEAD (BIT #13)
1758    040000          TAP=    40000           ,TAPE DRIVE (BIT #14)
1759    100000          NBA=    100000          ,NOT BLOCK ADDRESSED (B T #15)
```

```
1760
1761                            ;LOOK-AHEAD REGISTER (RPLA) (#07)
1762
1763    000001                  EXT1=   1               ;EXTENSION 1 (BIT #0)
1764    000002                  EXT2=   2               ;EXTENSION 2 (BIT #1)
1765    000004                  EXT4=   4               ;EXTENSION 3 (BIT #2)
1766    000010                  EXT10=  10              ;EXTENSION 4 (BIT #3)
1767    000020                  EXT20=  20              ;EXTENSION 5 (BIT #4)
1768    000040                  EXT40=  40              ;EXTENSION 6 (BIT #5)
1769    000100                  SC1=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
1770    000200                  SC2=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
1771                            ;SC4=   400             ;SECTOR COUNT FIELD 2 (BIT #8)
1772    001000                  SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
1773    002000                  SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
1774    004000                  TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
1775    010000                  TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
1776    020000                  TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
1777    040000                  TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
1778    100000                  TRK20=  100000          ;TRACK FIELD 5 (BIT #15)
1779
1780                            ;RP04 ERROR REGISTER #2 (RPER2) (#10)
1781
1782    000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
1783    000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
1784    000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
1785    000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
1786    000020                  MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
1787    000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
1788    000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
1789    000200                  FEN=    200             ;FAILSAFE ENABLED (BIT #7)
1790    000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
1791    001000                  MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
1792    002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
1793    004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
1794    010000                  VU30=   10000           ;30VOLT UNSAFE (BIT #12)
1795    020000                  PLU=    20000           ;PLO UNSAFE (BIT #13)
1796    100000                  ACU=    100000          ;AC UNSAFE (BIT #15)
1797
1798                            ;RP05/6 ERROR REGISTER #02 (RPER2) (#10)
1799
1800    000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
1801    000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
1802    000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
1803    000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
1804    000020                  RAW=    20              ;READ AND WRITE (BIT #4)
1805    000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
1806    000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
1807    000200                  ABS=    200             ;ABNORMAL STOP (BIT #7)
1808    000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
1809    001000                  MHS=    1000            ;MUTLTIPLE HEAD SELECT (BIT #9)
1810    002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
1811    004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
1812    020000                  PLU=    20000           ;PLO UNSAFE (BIT #12)
1813
1814                            .OFFSET REGISTER (RPOF) (#11)
1815
```

```
1816    000001          OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
1817    000002          OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
1818    000004          OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
1819    000010          OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
1820    000020          OF400=  20              ;OFFSET 400 MICRO INCHES (BIT #4)
1821    000040          OF800=  40              ;OFFSET 800 MICRO INCHES (BIT #5)
1822    000200          OFREV=  200             ;OFFSET NEGATIVE (REVERSE) (BIT #5)
1823    002000          HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
1824    004000          ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11)
1825    010000          FMT22=  10000           ;FORMAT BIT (BIT #12
1826
1827                    ;DESIRED CYLINDER ADDRESS (RPCA) (#12)
1828                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1829
1830                    ;CURRENT CYLINDER ADDRESS (RPCC) (#13)
1831                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1832
1833                    ;SERIAL NUMBER REGISTER (RPSN) (#14)
1834                    ;(EACH IS CALLED BY BIT NUMBER)
1835
1836                    ;RPO4 ERROR REGISTER #03 (RPER3) (#15)
1837
1838    000001          PSU=    1               ;PACK SPEED UNSAFE (BIT #0)
1839    000002          VUF=    2               ;VELOCITY UNSAFE (BIT #1)
1840    000010          UWR=    10              ;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
1841    000040          ACL=    40              ;AC LOW (BIT #5)
1842    000100          DCL=    100             ;DC LOW (BIT #6)
1843    040000          SKI=    40000           ;SEEK INCOMPLETE (BIT #14)
1844    100000          OCYL=   100000          ;OFF CYLINDER (BIT #15)
1845
1846                    ;RP05/6 ERROR REGISTER #03 (RPER3) (#15)
1847
1848    000001          DCU=    1               ;DC UNSAFE (BIT #0)
1849    000002          WAO=    2               ;WRITE AND OFFSET (BIT #1)
1850    000040          ACL=    40              ;AC LOW (BIT #5)
1851    000100          DCL=    100             ;DC LOW (BIT #6)
1852    020000          OPE=    20000           ;OPERATOR PLUG ERROR (BIT #13)
1853    040000          SKI=    40000            SEEK INCOMPLETE (BIT #14)
1854    100000          OCYL=   100000          ;OFF CYLINDER ERROR (BIT #15)
1855
1856                    ;ECC POSITION REGISTER (RPEC1) (#16)
1857                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1858
1859                    ;ECC PATTERN REGISTER (RPEC2) (#17)
1860                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1861
1862                    ;;*****************************************************************
1863
1864                    ;OP CODE DEFINITIONS
1865    000101          NOOP=101
1866    000103          UNLOAD=103
1867    000105          SEEK=105
1868    000107          RECAL=107
1869    000111          DRVCLR=111
1870    000113          RELEASE=113
1871    000115          OFFSET=115
```

```
1872            000117                  RTC=117
1873            000121                  READIN=121
1874            000123                  PACK=123
1875            000131                  SEARCH=131
1876            000151                  WRCKD=151
1877            000153                  WRCKHD=153
1878            000161                  WRITE=161
1879            000163                  WRTHD=163
1880            000171                  READ=171
1881            000173                  READHD=173
1882            000141                  GETREG=141
1883            000143                  SETFORM=143
1884            000145                  SELDRV=145
1885
1886                                    ;OTHER EQUATES
1887
1888            177400                  SCTRWC = -256                   ;WORD COUNT FOR SECTOR
1889            010000                  FMT22=10000                     ;FORMAT 22 BIT
1890
```

```
                                .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
```

```
1947            000012                  LF      =       12
1948  001220    000000                  C.SWR:  .WORD   0               ;CONTROL SWITCHES
1949  001222    000000                  SAVCSW: .WORD   0               ;PREVIOUS CONTENTS OF 'C.SWR'
1950  001224    000000                  CNTRLC: .WORD   0               ;CONTROL "C" FLAG
1951  001226    000000                  BUSADR: .WORD   0               ;GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
1952  001230    000000                  LPTAVL: .WORD   0               ;LPT AVAILABLE STATUS (0=NO,1=YES)
1953  001232    000000                  DRVSEL: .WORD   0               ;DRIVES SELECTED FOR TESTING
1954  001234    037777  000000          TSTNMS: .WORD   37777,0         ;RUN TESTS 0-15
1955  001240    000000  000000          OPNFLG: .WORD   0,0             ;MODIFY TEST PARAMETER FLAGS
1956  001244    000000                  CLKSTA: .WORD   0               ;CLOCK STATUS (0=NO CLOCK,+1=KW11-P,
1957                                                                    ;AND -1=KW11-L)
1958  001246    000020                  TICKMS: .WORD   16              ;16 MILLISECONDS PER CLOCK TICK
1959  001250    040432                  TICKUS: WORD    16666           ;16666 MIRCOSECONDS PER CLOCK TICK
1960  001252    000000                  BYPASS: .WORD   0
1961  001254    000000                  CHKDRV: .WORD   0               ;DRIVE UNDER TEST
1962  001256    000000                  DRVMSK: .WORD   0               ,DRIVE MASK BIT
1963  001260    000000                  SVSTAT: .WORD   0               ;STATUS/ERROR INDICATOR IS
1964                                                                    ;SAVED HERE ON AN ERROR
1965  001262    000000                  CYL.RD: .WORD   0               ;CYLINDER READ
1966  001264    000000                  TRK RD  .WORD   0               ;TRACK READ
1967  001266    000000                  SEC.RD  .WORD   0               ;SECTOR READ
1968  001270    000000                  CYL DS  .WORD   0               ;CYLINDER DESIRED
1969  001272    000000                  SEC DS: .WORD   0               ,SECTOR DESIRED
1970  001274    000000                  TRK.DS: .WORD   0               ;TRACK DESIRED
1971  001276    000000                  TIM.UP: .WORD   0               ;MINIMUM TIME
1972  001300    000000                          .WORD   0               ;NUMBER OF COUNTS BELOW MIN  LIMIT
1973  001302    000000                          .WORD   0               ;MAXIMUM TIME
1974  001304    000000                          .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
1975  001306    000000  000000                  .WORD   0,0             ,TOTAL TIME OF ALL SEEKS
1976  001312    000000                          WORD    0               ,NUMBER OF SEEKS PERFORMED
1977  001314    000000                  TIM DN  .WORD   0               ;MINIMUM TIME
1978  001316    000000                          WORD    0               ;NUMBER OF COUNTS BELOW MIN  LIMIT
1979  001320    000000                          .WORD   0               ;MAXIMUM TIME
1980  001322    000000                          .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
1981  001324    000000  000000                  .WORD   0,0             ;TOTAL TIME OF ALL SEEKS
1982  001330    000000                          WORD    0               ;NUMBER OF SEEKS PERFORMED
1983  001332    000000                  TIM.PT. .WORD   0               ;POINTS TO TABLE OF TIMES
1984  001334    000000                  WCEFLG: .WORD   0               ;FATAL WRITE CHECK ERROR FLAG (TEST 20)
1985  001336    000000                  STALLO: .WORD   0               ;VARIABLE STALL (TEST 4)
1986  001340    000000  000000          SVADR:  .WORD   0,0             ;SAVE DISK ADDRESS (TEST 22)
1987  001344    000000                  SEKTMR: .WORD   0               ;SEEK TIMER  (TEST 10)
1988  001346    000000                  SEKCNT: .WORD   0               ;SEEK COUNTER
1989  001350    000000                  DELTA:  .WORD   0               ;TESTING RANGE FOR SERVO SETTLE DOWN TEST
1990  001352    165000                  TRCKWC: .WORD   -(256 *22 )      ;WORD COUNT FOR A FULL TRACK IN 16 BIT MODE
1991  001354    000012                  STALL1: .WORD   10.             ;10 MILLISECONDS STALL (TEST 0-11)
1992  001356    000012                  STALL2: .WORD   10              ;10 MILLISECONDS STALL (TEST 16-21)
1993  001360    011610                  STALL3: .WORD   5000.           ;5 SEC STALL (TEST 22)
1994  001362    000031                  MXSTAL: .WORD   25              ,MAX. INCREMENTING STALL ALLOWED IN TEST 4
1995  001364      144                   ERR CT. .BYTE   100             ;NUMBER OF ERRORS ALLOWED IN TESTS 16 - 21
1996                                                                    ;BEFORE GOING TO THE NEXT TEST
1997  001365      000                           .BYTE   0               ;RESERVED
1998
1999                                  ;ADDRESSES AND VECTORS
2000  001366    176700                  RH.ADR. .WORD   176700          ,RH11-RH70 UNIBUS ADDRESS
2001  001370    000254  000240          RHVEC:  WORD    254,5*32        ,RH11-RH70 VECTOR ADDRESS AND PRIORITY
2002  001374    000104  000106          PKV     WORD    104,106         ,KW11-P VECTOR ADDRESS
```

```
2003  001400  172540          PKCS:   .WORD   172540          ;KW11-P CONTROL AND STATUS REG.
2004  001402  172542          PKB:    .WORD   172542          ;KW11-P COUNT SET BUFFER
2005  001404  172544          PKC:    .WORD   172544          ;KW11-P COUNTER
2006  001406  000100  000102  LKV:    .WORD   100,102         ;KW11-L VECTOR ADDRESS
2007  001412  177546          LKS:    .WORD   177546          ;KW11-L STATUS REGISTER
2008  001414  177564          TPS     .WORD   177564          ;TTY PRINTER STATUS
2009  001416  177566          TPB     .WORD   177566          ;TTY PRINTER BUFFER
2010  001420  177514          LPS     .WORD   177514          ;LINE PRINTER STATUS
2011  001422  177516          LPB:    .WORD   177516          ;LINE PRINTER BUFFER
2012
2013                          ,BIT TABLE
2014  001424  000001          BITS.   .WORD   BIT00
2015  001426  000002                  .WORD   BIT01
2016  001430  000004                  .WORD   BIT02
2017  001432  000010                  .WORD   BIT03
2018  001434  000020                  .WORD   BIT04
2019  001436  000040                   WORD   BIT05
2020  001440  000100                  .WORD   BIT06
2021  001442  000200                  .WORD   BIT07
2022  001444  000400                  .WORD   BIT08
2023  001446  001000                  .WORD   BIT09
2024  001450  002000                  .WORD   BIT10
2025  001452  004000                  .WORD   BIT11
2026  001454  010000                  .WORD   BIT12
2027  001456  020000                  .WORD   BIT13
2028  001460  040000                  .WORD   BIT14
2029  001462  100000                  .WORD   BIT15
2030  001464  000001                  .WORD   BIT00
2031  001466  000002                  .WORD   BIT01
2032  001470  000004                  .WORD   BIT02
2033  001472  000010                  .WORD   BIT03
2034  001474  000020                  .WORD   BIT04
2035  001476  000040                  .WORD   BIT05
2036  001500  000100                   WORD   BIT06
2037  001502  000200                   WORD   BIT07
2038
2039
2040                          ;COMMON STORAGE FOR TEST PARAMETER
2041  001504  000000          PRM:    .WORD   0
2042  001506  000000          RPT     .WORD   0               ;REPEAT COUNTS FOR ALL TESTS
2043  001510  000000          FC       WORD   0               ,FIRST CYLINDER
2044  001512  000000          LC       WORD   0               ,LAST CYLINDER
2045  001514  000000          IC      .WORD   0               ,INCREMENT CYLINDER
2046  001516  000000          FT      .WORD   0               ,FIRST TRACK
2047  001520  000000          LT       WORD   0               ;LAST TRACK
2048  001522  000000          IT       WORD   0               ,INCREMENT TRACK
2049  001524  000000          FS.     .WORD   0               ,FIRST SECTOR
2050  001526  000000          LS      .WORD   0               ,LAST SECTOR
2051  001530  000000          PAT.    .WORD   0               ,PATTERN CODE
2052  001532  000000          NC1     .WORD   0               ,NEW CYLINDER ADDRESS
2053  001534  000000          NC2      WORD   0               ,NEW CYLINDER ADDRESS
2054
2055                          ,TABLE OF PARAMETER POINTERS
2056  001536  002330          PRMPT   .WORD   PRM0
2057  001540  002344                  .WORD   PRM1
2058  001542  002372                   WORD   PRM2
```

```
 2059  001544  002414                          .WORD   PRM3
 2060  001546  002436                          .WORD   PRM4
 2061  001550  002460                          .WORD   PRM5
 2062  001552  002502                          .WORD   PRM6
 2063  001554  002524                          .WORD   PRM7
 2064  001556  002544                          .WORD   PRM10
 2065  001560  002564                          .WORD   PRM11
 2066  001562  002606                          .WORD   PRM12
 2067  001564  002622                          .WORD   PRM13
 2068  001566  002636                          .WORD   PRM14
 2069  001570  002652                          .WORD   PRM15
 2070  001572  002666                          .WORD   PRM16
 2071  001574  002700                          .WORD   PRM17
 2072  001576  002712                          .WORD   PRM20
 2073  001600  002744                          .WORD   PRM21
 2074  001602  002760                           WORD   PRM22
 2075  001604  000000                           WORD   0                ,TERMINATOR
 2076
 2077                                  ,PARAMETER UPPER LIMIT
 2078  001606  032767                  PRMLMT:  .WORD   32767           ,"R"
 2079  001610  000632                           .WORD   410             ,"FC"
 2080  001612  000632                           .WORD   410.            ,"LC"
 2081  001614  001456                           .WORD   814             ,"FC'"
 2082  001616  001456                           .WORD   814             ,"LC'"
 2083  001620  001456                           .WORD   814.            ,"IC"
 2084  001622  000022                           .WORD   18.             ,"FT"
 2085  001624  000022                           .WORD   18              ,"LT"
 2086  001626  000022                            WORD   18.             ,"IT"
 2087  001630  000025                           .WORD   21              ,"FS"
 2088  001632  000025                           .WORD   21.             ,"LS"
 2089  001634  177777                           .WORD   177777          ,"PAT"
 2090
 2091                                  ,TABLE OF MESSAGE POINTERS
 2092  001636  043000                  PRMMSG:  .WORD   MSG.R
 2093  001640  043002                           .WORD   MSG FC
 2094  001642  043005                            WORD   MSG LC
 2095  001644  043010                           .WORD   MSGFCP
 2096  001646  043014                           .WORD   MSGLCP
 2097  001650  043020                           .WORD   MSG.IC
 2098  001652  043023                           .WORD   MSG FT
 2099  001654  043026                           .WORD   MSG.LT
 2100  001656  043031                           .WORD   MSG.IT
 2101  001660  043034                            WORD   MSG FS
 2102  001662  043037                           .WORD   MSG.LS
 2103
 2104                                  ,STATUS/ERROR INDICATOR MESSAGES POINTER TABLE
 2105                                  ,DEFAULT VALUES OF TEST PARAMETERS
 2106  001664  001125  000310  000632  DFLT.    .WORD   1125,200 ,410 ,814 ,0,0 ;RECAL/SEEK (T0)
 2107  001672  001456  000000  000000
 2108  001700  003377  000144  000000           .WORD   3377,100 ,0,128 ,0,256 ,0,0,0,0,0 ,SEEK/SEEK (T1)
 2109  001706  000200  000000  000400
 2110  001714  000000  000000  000000
 2111  001722  000000  000000
 2112  001726  001177  000001  000000            WORD   1177,1,0,410 ,0,814 ,1,0,0 ,INCREMENT SEEK (T2)
 2113  001734  000632  000000  001456
 2114  001742  000001  000000  000000
```

```
2115  001750  001177  000010  000000          .WORD   1177,10,0,256.,0,256.,1,0,0 ;STEPPING SEEK (T3)
2116  001756  000400  000000  000400
2117  001764  000001  000000  000000
2118  001772  001177  000001  000000           WORD   1177,1,0,410.,0,814.,1,0,0 ;OSCILLATING SEEK (T4)
2119  002000  000632  000000  001456
2120  002006  000001  000000  000000
2121  002014  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;CONVERGING/DIVERGING SEEK (T5)
2122  002022  000632  000000  001456
2123  002030  000001  000000  000000
2124  002036  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;SERVO ADDRESSING LOGIC NOISE (T6)
2125  002044  000632  000000  001456
2126  002052  000001  000000  000000
2127  002060  000337  011610  000000          .WORD   337,5000.,0,410.,0,814.,0,18.   ;RANDOM SEEK TEST (T7)
2128  002066  000632  000000  001456
2129  002074  000000  000022
2130  002100  000177  000001  000000           WORD   177,1,0,410.,0,814.,100.,0  ;SERVO SETTLE DOWN TEST (T10)
2131  002106  000632  000000  001456
2132  002114  000144  000000
2133  002120  001177  000001  000000           WORD   1177,1,0,410.,0,814.,1,0,0      ,ALL SEEKS TEST (T11)
2134  002126  000632  000000  001456
2135  002134  000001  000000  000000
2136  002142  001113  000001  000000           WORD   1113,1,0,0,0,0  ;ROTATIONAL SPEED TIMING TEST (T12)
2137  002150  000000  000000  000000
2138  002156  000037  000001  000000           WORD   37,1,0,410.,0,814.        ;ONE CYLINDER SEEK TIMING TEST (T13)
2139  002164  000632  000000  001456
2140  002172  000037  000001  000000           WORD   37,1,0,136.,0,255.        ;ACCESS TIME MEASURMENT TEST (T14)
2141  002200  000210  000000  000377
2142  002206  000037  000001  000000          .WORD   37,1,0,410.,0,814.        ;MAXIMUM SEEK TIMING TEST (T15)
2143  002214  000632  000000  001456
2144  002222  000113  000001  000000           WORD   113,1,0,0,0      ,SECTOR ADDRESSING TEST (T16)
2145  002230  000000  000000
2146  002234  001013  000001  000000           WORD   1013,1,0,0,0     ;TRACK ADDRESSING TEST (T17)
2147  002242  000000  000000
2148  002246  007777  000001  000000           WORD   7777,1,0,410.,0,814.,64.,0,18.,1,1,0,177777 ,DATA TEST (T20)
2149  002254  000632  000000  001456
2150  002262  000100  000000  000022
2151  002270  000001  000001  000000
2152  002276  177777
2153  002300  000037  047040  000000          .WORD   37,20000.,0,410.,0,814.  ,EXERCISER (T21)
2154  002306  000632  000000  001456
2155  002314  000037  011610  000000          .WORD   37,5000.,0,136.,0,255    ,RP04 ACCESS TIME ADJUSTMENT TEST (T22)
2156  002322  000210  000000  000377
2157
2158                                  ;PARAMETER TABLES
2159
2160                                  ;RECAL/SEEK (T0)
2161  002330  001125                  PRM0    .WORD   1125
2162  002332  000310                          WORD    200
2163  002334  000632                          WORD    410
2164  002336  001456                          .WORD   814
2165  002340  000000                          .WORD   0
2166  002342  000000                          WORD    0
2167
2168                                  ;SEEK/SEEK (T1)
2169  002344  003377                  PRM1    WORD    3377
2170  002346  000144                          WORD    100
```

```
 2171  002350  000000                    .WORD   0
 2172  002352  000200                    .WORD   128.
 2173  002354  000000                    .WORD   0
 2174  002356  000400                    .WORD   256.
 2175  002360  000000                    .WORD   0
 2176  002362  000000                     WORD   0
 2177  002364  000000                    .WORD   0
 2178  002366  000000                    .WORD   0
 2179  002370  000000                    .WORD   0
 2180
 2181                              ;INCREMENT SEEK (T2)
 2182  002372  001177              PRM2.   .WORD   1177
 2183  002374  000001                    .WORD   1
 2184  002376  000000                    .WORD   0
 2185  002400  000632                    .WORD   410
 2186  002402  000000                    .WORD   0
 2187  002404  001456                     WORD   814
 2188  002406  000001                     WORD   1
 2189  002410  000000                    .WORD   0
 2190  002412  000000                    .WORD   0
 2191
 2192                              ;STEPPING SEEK (T3)
 2193  002414  001177              PRM3    .WORD   1177
 2194  002416  000001                     WORD   1
 2195  002420  000000                     WORD   0
 2196  002422  000400                    .WORD   256.
 2197  002424  000000                    .WORD   0
 2198  002426  001000                     WORD   512
 2199  002430  000001                    .WORD   1
 2200  002432  000000                     WORD   0
 2201  002434  000000                     WORD   0
 2202
 2203                              ;OSCILLATING SEEK (T4)
 2204  002436  001177              PRM4     WORD   1177
 2205  002440  000001                     WORD   1
 2206  002442  000000                     WORD   0
 2207  002444  000632                     WORD   410
 2208  002446  000000                     WORD   0
 2209  002450  001456                     WORD   814
 2210  002452  000001                     WORD   1
 2211  002454  000000                    .WORD   0
 2212  002456  000000                    .WORD   0
 2213
 2214                              ;CONVERGING/DIVERGING SEEK (T5)
 2215  002460  001177              PRM5    .WORD   1177
 2216  002462  000001                     WORD   1
 2217  002464  000000                    .WORD   0
 2218  002466  000632                    .WORD   410
 2219  002470  000000                    .WORD   0
 2220  002472  001456                    .WORD   814
 2221  002474  000001                     WORD   1
 2222  002476  000000                    .WORD   0
 2223  002500  000000                     WORD   0
 2224
 2225                              ;SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)
 2226  002502  001177              PRM6     WORD   1177
```

```
2227  002504  000001                    .WORD   1
2228  002506  000000                    .WORD   0
2229  002510  000632                     WORD   410.
2230  002512  000000                    .WORD   0
2231  002514  001456                    .WORD   814.
2232  002516  000001                    .WORD   1
2233  002520  000000                     WORD   0
2234  002522  000000                    .WORD   0
2235
2236                              ;RANDOM SEEK TEST (T7)
2237  002524  000337              PRM7  .WORD   337
2238  002526  011610                    .WORD   5000.
2239  002530  000000                    .WORD   0
2240  002532  000632                    .WORD   410.
2241  002534  000000                     WORD   0
2242  002536  001456                     WORD   814
2243  002540  000000                     WORD   0
2244  002542  000022                    .WORD   18
2245
2246                              ;SERVO SETTLE DOWN TEST (T10)
2247  002544  000177              PRM10 .WORD   177
2248  002546  000001                     WORD   1
2249  002550  000000                    .WORD   0
2250  002552  000632                    .WORD   410
2251  002554  000000                     WORD   0
2252  002556  001456                    .WORD   814.
2253  002560  000144                    .WORD   100.
2254  002562  000000                    .WORD   0
2255
2256                              ;ALL SEEKS TEST (T11)
2257  002564  001177              PRM11 .WORD   1177
2258  002566  000001                    .WORD   1
2259  002570  000000                    .WORD   0
2260  002572  000632                    .WORD   410.
2261  002574  000000                     WORD   0
2262  002576  001456                     WORD   814
2263  002600  000001                     WORD   1
2264  002602  000000                    .WORD   0
2265  002604  000000                    .WORD   0
2266
2267                              ;ROTATIONAL SPEED TIMING TEST (T12)
2268  002606  001113              PRM12  WORD   1113
2269  002610  000001                     WORD   1
2270  002612  000000                     WORD   0
2271  002614  000000                     WORD   0
2272  002616  000000                     WORD   0
2273  002620  000000                     WORD   0
2274
2275                              ;ONE CYLINDER SEEK TIMING TEST (T13)
2276  002622  000037              PRM13 .WORD   37
2277  002624  000001                    .WORD   1
2278  002626  000000                     WORD   0
2279  002630  000632                     WORD   410
2280  002632  000000                     WORD   0
2281  002634  001456                     WORD   814
2282
```

```
2283                                    ;ACCESS TIME MEASURMENT TEST (T14)
2284  002636  000037                    PRM14:  .WORD   37
2285  002640  000001                            .WORD   1
2286  002642  000000                            WORD    0
2287  002644  000210                            WORD    136.
2288  002646  000000                            .WORD   0
2289  002650  000377                            WORD    255.
2290
2291                                    ,MAXIMUM SEEK TIMING TEST (T15)
2292  002652  000037                    PRM15   .WORD   37
2293  002654  000001                            WORD    1
2294  002656  000000                            WORD    0
2295  002660  000632                            WORD    410
2296  002662  000000                            WORD    0
2297  002664  001456                            WORD    814
2298
2299                                    ,SECTOR ADDRESSING TEST (T16)
2300  002666  000113                    PRM16   WORD    113
2301  002670  000001                            .WORD   1
2302  002672  000000                            WORD    0
2303  002674  000000                            .WORD   0
2304  002676  000000                            WORD    0
2305
2306                                    ,TRACK ADDRESSING TEST (T17)
2307  002700  001013                    PRM17   .WORD   1013
2308  002702  000001                            WORD    1
2309  002704  000000                            WORD    0
2310  002706  000000                            .WORD   0
2311  002710  000000                            WORD    0
2312
2313                                    ,DATA TEST (T20)
2314  002712  007777                    PRM20   WORD    7777
2315  002714  000001                            WORD    1
2316  002716  000000                            WORD    0
2317  002720  000632                            WORD    410
2318  002722  000000                            WORD    0
2319  002724  001456                            WORD    814
2320  002726  000100                            WORD    64
2321  002730  000000                            WORD    0
2322  002732  000022                            WORD    18
2323  002734  000001                            WORD    1
2324  002736  000001                            WORD    1
2325  002740  000000                            WORD    0
2326  002742  177777                    PTRN15  WORD    177777
2327
2328                                    ,EXERCISER (T21)
2329  002744  000037                    PRM21   WORD    37
2330  002746  047040                            WORD    20000
2331  002750  000000                            WORD    0
2332  002752  000632                            WORD    410
2333  002754  000000                            WORD    0
2334  002756  001456                            WORD    814
2335
2336                                    ,RP04 ACCESS TIME ADJUSTMENT TEST (T22)
2337  002760  000037                    PRM22   WORD    37
2338  002762  011610                            WORD    5000
```

```
 2339  002764  000000                    .WORD   0
 2340  002766  000210                     WORD   136.
 2341  002770  000000                     WORD   0
 2342  002772  000377                    .WORD   255.
 2343
 2344
 2345
 2346                          ;SEEK TIMING LIMITS
 2347
 2348  002774  043420          T7A       WORD   MSG7
 2349  002776  000000                     WORD   0
 2350  003000  003142                     WORD   1634          ;(16 67-2%)**2
 2351  003002  003244                     WORD   1700          ;(16 67+2%)**2
 2352
 2353  003004  043420          T7B       WORD   MSG7
 2354  003006  000000                     WORD   0
 2355  003010  003131                     WORD   1625          ;(16 67-2 5%)**2
 2356  003012  003255                     WORD   1709          ;(16 67+2 5%)**2
 2357
 2358  003014  043452          T10       WORD   MSG10A
 2359  003016  043521                     WORD   MSG10B
 2360  003020  000000                     WORD   0             ;NO LOWER LIMIT
 2361  003022  001750                     WORD   1000          ;(7+3)**2
 2362
 2363  003024  043536          T11       WORD   MSG11A
 2364  003026  043604                     WORD   MSG11B
 2365  003030  000000                     WORD   0             ;NO LOWER LIMIT
 2366  003032  005670                     WORD   3000          ;(28+2)**2
 2367
 2368  003034  043621          T12       WORD   MSG12A
 2369  003036  043663                     WORD   MSG12B
 2370  003040  000000                     WORD   0             ;NO LOWER LIMIT
 2371  003042  012574                     WORD   5500          ;(50+2)**2
 2372
 2373  003044  003104          PAT PT    WORD   PAT0          ;TABLE OF POINTERS WHICH POINT TO THE
 2374  003046  003144                     WORD   PAT1          ;PATTERNS USED BY THE DATA TEST
 2375  003050  003204                     WORD   PAT2
 2376  003052  003244                    .WORD   PAT3
 2377  003054  003304                     WORD   PAT4
 2378  003056  003344                     WORD   PAT5
 2379  003060  003404                     WORD   PAT6
 2380  003062  003444                     WORD   PAT7
 2381  003064  003504                     WORD   PAT8
 2382  003066  003544                     WORD   PAT9
 2383  003070  003604                     WORD   PAT10
 2384  003072  003644                     WORD   PAT11
 2385  003074  003704                     WORD   PAT12
 2386  003076  003744                     WORD   PAT13
 2387  003100  004004                     WORD   PAT14
 2388  003102  004044                     WORD   PAT15
 2389
 2390                          ;PATTERNS 0 THRU 15
 2391
 2392  003104  165555          PAT0      WORD   165555        ;PATTERN 0
 2393  003106  133333                     WORD   133333
 2394  003110  165555                     WORD   165555
```

```
 2395  003112  133333                    .WORD   133333
 2396  003114  165555                     WORD   165555
 2397  003116  133333                    .WORD   133333
 2398  003120  165555                     WORD   165555
 2399  003122  133333                     WORD   133333
 2400  003124  165555                    .WORD   165555
 2401  003126  133333                    .WORD   133333
 2402  003130  165555                     WORD   165555
 2403  003132  133333                    .WORD   133333
 2404  003134  165555                    .WORD   165555
 2405  003136  133333                     WORD   133333
 2406  003140  165555                     WORD   165555
 2407  003142  133333                     WORD   133333
 2408
 2409  003144  000001              PAT1   .WORD   000001          ,PATTERN 1
 2410  003146  000003                     WORD   000003
 2411  003150  000007                     WORD   000007
 2412  003152  000017                     WORD   000017
 2413  003154  000037                     WORD   000037
 2414  003156  000077                     WORD   000077
 2415  003160  000177                     WORD   000177
 2416  003162  000377                     WORD   000377
 2417  003164  000777                     WORD   000777
 2418  003166  001777                     WORD   001777
 2419  003170  003777                     WORD   003777
 2420  003172  007777                     WORD   007777
 2421  003174  017777                     WORD   017777
 2422  003176  037777                     WORD   037777
 2423  003200  077777                     WORD   077777
 2424  003202  177777                     WORD   177777
 2425
 2426  003204  177776              PAT2   .WORD   177776          ,PATTERN 2
 2427  003206  177774                     WORD   177774
 2428  003210  177770                     WORD   177770
 2429  003212  177760                     WORD   177760
 2430  003214  177740                     WORD   177740
 2431  003216  177700                     WORD   177700
 2432  003220  177600                     WORD   177600
 2433  003222  177400                     WORD   177400
 2434  003224  177000                     WORD   177000
 2435  003226  176000                     WORD   176000
 2436  003230  174000                     WORD   174000
 2437  003232  170000                     WORD   170000
 2438  003234  160000                     WORD   160000
 2439  003236  140000                     WORD   140000
 2440  003240  100000                     WORD   100000
 2441  003242  000000                     WORD   000000
 2442
 2443  003244  000000              PAT3   WORD   000000          ,PATTERN 3
 2444  003246  000000                     WORD   000000
 2445  003250  000000                     WORD   000000
 2446  003252  177777                     WORD   177777
 2447  003254  177777                     WORD   177777
 2448  003256  177777                     WORD   177777
 2449  003260  000000                     WORD   000000
 2450  003262  000000                     WORD   000000
```

```
                                                                H 4
2451  003264  177777                        .WORD   177777
2452  003266  177777                        .WORD   177777
2453  003270  000000                         WORD   000000
2454  003272  177777                        .WORD   177777
2455  003274  000000                        .WORD   000000
2456  003276  177777                        .WORD   177777
2457  003300  000000                        .WORD   000000
2458  003302  177777                         WORD   177777
2459
2460  003304  000000              PAT4      .WORD   000000          ;PATTERN 4
2461  003306  010421                         WORD   010421
2462  003310  021042                        .WORD   021042
2463  003312  031463                        .WORD   031463
2464  003314  042104                         WORD   042104
2465  003316  052525                         WORD   052525
2466  003320  063146                        .WORD   063146
2467  003322  073567                        .WORD   073567
2468  003324  104210                        .WORD   104210
2469  003326  114631                        .WORD   114631
2470  003330  125252                         WORD   125252
2471  003332  135673                         WORD   135673
2472  003334  146314                         WORD   146314
2473  003336  156735                         WORD   156735
2474  003340  167356                         WORD   167356
2475  003342  177777                         WORD   177777
2476
2477  003344  052525              PAT5       WORD   052525          ;PATTERN 5
2478  003346  052525                         WORD   052525
2479  003350  052525                         WORD   052525
2480  003352  125252                         WORD   125252
2481  003354  125252                         WORD   125252
2482  003356  125252                         WORD   125252
2483  003360  052525                         WORD   052525
2484  003362  052525                         WORD   052525
2485  003364  125252                         WORD   125252
2486  003366  125252                         WORD   125252
2487  003370  052525                         WORD   052525
2488  003372  125252                         WORD   125252
2489  003374  052525                         WORD   052525
2490  003376  125252                         WORD   125252
2491  003400  052525                         WORD   052525
2492  003402  125252                         WORD   125252
2493
2494  003404  007417              PAT6       WORD   007417          ;PATTERN 6
2495  003406  007417                         WORD   007417
2496  003410  007417                         WORD   007417
2497  003412  170360                        .WORD   170360
2498  003414  170360                         WORD   170360
2499  003416  170360                        .WORD   170360
2500  003420  007417                         WORD   007417
2501  003422  007417                        .WORD   007417
2502  003424  170360                         WORD   170360
2503  003426  170360                         WORD   170360
2504  003430  007417                         WORD   007417
2505  003432  170360                         WORD   170360
2506  003434  007417                         WORD   007417
```

```
 2507  003436  170360                    WORD   170360
 2508  003440  007417                    WORD   007417
 2509  003442  170360                    .WORD  170360
 2510
 2511  003444  026455          PAT7      .WORD  026455          ;PATTERN 7
 2512  003446  026455                    .WORD  026455
 2513  003450  026455                    WORD   026455
 2514  003452  151322                    WORD   151322
 2515  003454  151322                    .WORD  151322
 2516  003456  151322                    .WORD  151322
 2517  003460  026455                    .WORD  026455
 2518  003462  026455                    .WORD  026455
 2519  003464  151322                    .WORD  151322
 2520  003466  151322                    .WORD  151322
 2521  003470  026455                    .WORD  026455
 2522  003472  151322                    .WORD  151322
 2523  003474  026455                    .WORD  026455
 2524  003476  151322                    WORD   151322
 2525  003500  026455                    .WORD  026455
 2526  003502  151322                    WORD   151322
 2527
 2528  003504  165555          PAT8      WORD   165555          ;PATTERN 8
 2529  003506  133333                    WORD   133333
 2530  003510  165555                    WORD   165555
 2531  003512  133333                    .WORD  133333
 2532  003514  165555                    .WORD  165555
 2533  003516  133333                    .WORD  133333
 2534  003520  165555                    .WORD  165555
 2535  003522  133333                    WORD   133333
 2536  003524  165555                    WORD   165555
 2537  003526  133333                    .WORD  133333
 2538  003530  165555                    WORD   165555
 2539  003532  133333                    WORD   133333
 2540  003534  165555                    WORD   165555
 2541  003536  133333                    WORD   133333
 2542  003540  165555                    WORD   165555
 2543  003542  133333                    WORD   133333
 2544
 2545  003544  000001          PAT9      .WORD  000001          ;PATTERN 9
 2546  003546  000002                    WORD   000002
 2547  003550  000004                    WORD   000004
 2548  003552  000010                    .WORD  000010
 2549  003554  000020                    .WORD  000020
 2550  003556  000040                    WORD   000040
 2551  003560  000100                    WORD   000100
 2552  003562  000200                    .WORD  000200
 2553  003564  000400                    .WORD  000400
 2554  003566  001000                    .WORD  001000
 2555  003570  002000                    .WORD  002000
 2556  003572  004000                    .WORD  004000
 2557  003574  010000                    .WORD  010000
 2558  003576  020000                    .WORD  020000
 2559  003600  040000                    WORD   040000
 2560  003602  100000                    WORD   100000
 2561
 2562  003604  177776          PAT10     WORD   177776          ;PATTERN 10
```

```
2563  003606  177775                      WORD   177775
2564  003610  177773                     .WORD   177773
2565  003612  177767                     .WORD   177767
2566  003614  177757                     .WORD   177757
2567  003616  177737                     .WORD   177737
2568  003620  177677                     .WORD   177677
2569  003622  177577                     .WORD   177577
2570  003624  177377                      WORD   177377
2571  003626  176777                      WORD   176777
2572  003630  175777                     .WORD   175777
2573  003632  173777                     .WORD   173777
2574  003634  167777                     .WORD   167777
2575  003636  157777                      WORD   157777
2576  003640  137777                     .WORD   137777
2577  003642  077777                     .WORD   077777
2578
2579  003644  172666              PAT11   WORD   172666          ,PATTERN 11
2580  003646  155555                      WORD   155555
2581  003650  172666                     .WORD   172666
2582  003652  155555                     .WORD   155555
2583  003654  172666                     .WORD   172666
2584  003656  155555                     .WORD   155555
2585  003660  172666                     .WORD   172666
2586  003662  155555                     .WORD   155555
2587  003664  172666                      WORD   172666
2588  003666  155555                     .WORD   155555
2589  003670  172666                     .WORD   172666
2590  003672  155555                      WORD   155555
2591  003674  172666                     .WORD   172666
2592  003676  155555                      WORD   155555
2593  003700  172666                     .WORD   172666
2594  003702  155555                      WORD   155555
2595
2596  003704  077777              PAT12   WORD   077777          ,PATTERN 12
2597  003706  137777                      WORD   137777
2598  003710  157777                      WORD   157777
2599  003712  167777                     .WORD   167777
2600  003714  173777                     .WORD   173777
2601  003716  175777                     .WORD   175777
2602  003720  176777                     .WORD   176777
2603  003722  177377                     .WORD   177377
2604  003724  177577                     .WORD   177577
2605  003726  177677                     .WORD   177677
2606  003730  177737                     .WORD   177737
2607  003732  177757                      WORD   177757
2608  003734  177767                     .WORD   177767
2609  003736  177773                      WORD   177773
2610  003740  177775                     .WORD   177775
2611  003742  177776                     .WORD   177776
2612
2613  003744  153333              PAT13.  .WORD   153333          ,PATTERN 13
2614  003746  066667                      .WORD   066667
2615  003750  153333                      .WORD   153333
2616  003752  066667                      .WORD   066667
2617  003754  153333                      .WORD   153333
2618  003756  066667                      .WORD   066667
```

```
2619  003760  153333                    .WORD   153333
2620  003762  066667                    .WORD   066667
2621  003764  153333                    .WORD   153333
2622  003766  066667                    .WORD   066667
2623  003770  153333                    .WORD   153333
2624  003772  066667                    .WORD   066667
2625  003774  153333                    .WORD   153333
2626  003776  066667                    .WORD   066667
2627  004000  153333                    .WORD   153333
2628  004002  066667                    .WORD   066667
2629
2630  004004  000000            PAT14   .WORD   000000          ;PATTERN 14
2631  004006  177777                    .WORD   177777
2632  004010  177777                    .WORD   177777
2633  004012  177777                    .WORD   177777
2634  004014  177777                    .WORD   177777
2635  004016  177777                    WORD    177777
2636  004020  177777                    .WORD   177777
2637  004022  177777                    .WORD   177777
2638  004024  177777                    .WORD   177777
2639  004026  177777                    .WORD   177777
2640  004030  177777                    WORD    177777
2641  004032  177777                    .WORD   177777
2642  004034  177777                    .WORD   177777
2643  004036  177777                    WORD    177777
2644  004040  177777                    WORD    177777
2645  004042  177777                    .WORD   177777
2646
2647  004044  177777            PAT15·  .WORD   177777          ,PATTERN 15
2648  004046  000000                    .WORD   000000
2649  004050  000000                    .WORD   000000
2650  004052  000000                    WORD    000000
2651  004054  000000                    .WORD   000000
2652  004056  000000                    .WORD   000000
2653  004060  000000                    WORD    000000
2654  004062  000000                    WORD    000000
2655  004064  000000                    .WORD   000000
2656  004066  000000                    .WORD   000000
2657  004070  000000                    .WORD   000000
2658  004072  000000                    .WORD   000000
2659  004074  000000                    .WORD   000000
2660  004076  000000                    .WORD   000000
2661  004100  000000                    .WORD   000000
2662  004102  000000                    .WORD   000000
2663
2664                          ,DPB (DATA PARAMETER BLOCK)
2665
2666  004104      000       DPB A   .BYTE   0               ,(0) DRIVE NUMBER
2667  004105      000               BYTE    0               ,(1) OFFSET VALUE OR FMT22, ECI, AND HCI
2668  004106      000               .BYTE   0               ,(2) COMMAND
2669  004107      000               .BYTE   0               ,(3) PSEL AND A17 AND A16
2670  004110  000000                WORD    0               ,(4) WORD COUNT (MUST BE NEG )
2671  004112  047712                WORD    BUFFER          ,(6) BUFFER ADDRESS OR
2672                                                        ,REGISTER TABLE POINTER
2673  004114      000               BYTE    0               ,(10) SECTOR ADDRESS OR
2674                                                        ,FIRST REG  INDEX
```

```
2675  004115  000                       BYTE   0              ;(11) TRACK ADDRESS OR
2676                                                           ;LAST REG.  INDEX
2677  004116  000000                    WORD   0              ;(12) CYLINDER ADDRESS
2678  004120  004204                    WORD   RP.REG         ;(14) ERROR TABLE POINTER
2679                                                           ;POINTS TO THE FIRST OF TWENTY
2680                                                           ;LOCATIONS OF WHERE THE DRIVER
2681                                                           ;IS TO STORE THE RH11/RP04
2682                                                           ;REGISTERS ON AN ERROR.  IF LEFT
2683                                                           ;ZERO REGISTERS ARE NOT SAVED.
2684  004122  000000                    WORD   0              ;(16) STATUS/ERROR INDICATOR
2685                                                           ;BIT15=1=>ERROR OCCURRED
2686                                                           ;BIT07=1=>DONE
2687                                                           ;BIT14-BIT09 AND BIT06-BIT03
2688                                                           ;INDICATE TYPE OF ERROR
2689
2690  004124  000             DPB B     BYTE   0              ;(0) DRIVE NUMBER
2691  004125  000                       BYTE   0              ;(1) OFFSET VALUE OR FMT22, ECI, AND HCI
2692  004126  000                       BYTE   0              ;(2) COMMAND
2693  004127  000                      .BYTE   0              ;(3) PSEL AND A17 AND A16
2694  004130  177776                    WORD   -2             ;(4) WORD COUNT (MUST BE NEG )
2695  004132  047712                    WORD   BUFFER         ,(6) BUFFER ADDRESS OR
2696                                                           ;REGISTER TABLE POINTER
2697  004134  000                      .BYTE   0              ,(10) SECTOR ADDRESS OR
2698                                                           ;FIRST REG.  INDEX
2699  004135  000                       BYTE   0              ,(11) TRACK ADDRESS OR
2700                                                           ,LAST REG.  INDEX
2701  004136  000000                    WORD   0              ;(12) CYLINDER ADDRESS
2702  004140  004204                    WORD   RP REG         ,(14) ERROR TABLE POINTER
2703                                                           ;POINTS TO THE FIRST OF TWENTY
2704                                                           ;LOCATIONS OF WHERE THE DRIVER
2705                                                           ;IS TO STORE THE RH11/RP04
2706                                                           ;REGISTERS ON AN ERROR.  IF LEFT
2707                                                           ;ZERO REGISTERS ARE NOT SAVED
2708  004142  000000                    WORD   0              ,(16) STATUS/ERROR INDICATOR
2709                                                           ;BIT15=1=>ERROR OCCURRED
2710                                                           ,BIT07=1=>DONE
2711                                                           ,BIT14-BIT09 AND BIT06-BIT03
2712                                                           ,INDICATE TYPE OF ERROR
2713
2714  004144  000             DPB C    .BYTE   0              ,(0) DRIVE NUMBER
2715  004145  000                      .BYTE   0              ,(1) OFFSET VALUE OR FMT22, ECI, AND HCI
2716  004146  000                       BYTE   0              ,(2) COMMAND
2717  004147  000                      .BYTE   0              ,(3) PSEL AND A17 AND A16
2718  004150  177776                    WORD   -2             ,(4) WORD COUNT (MUST BE NEG )
2719  004152  047712                    WORD   BUFFER         ,(6) BUFFER ADDRESS OR
2720                                                           ;REGISTER TABLE POINTER
2721  004154  000                       BYTE   0              ,(10) SECTOR ADDRESS OR
2722                                                           ,FIRST REG.  INDEX
2723  004155  000                       BYTE   0              ;(11) TRACK ADDRESS OR
2724                                                           ;LAST REG.  INDEX
2725  004156  000000                    WORD   0              ;(12) CYLINDER ADDRESS
2726  004160  004204                    WORD   RP REG         ,(14) ERROR TABLE POINTER
2727                                                           ;POINTS TO THE FIRST OF TWENTY
2728                                                           ,LOCATIONS OF WHERE THE DRIVER
2729                                                           ;IS TO STORE THE RH11/RP04
2730                                                           ,REGISTERS ON AN ERROR  IF LEFT
```

```
2731
2732  004162  000000                        WORD   0              ;ZERO REGISTERS ARE NOT SAVED.
2733                                                              ;(16) STATUS/ERROR INDICATOR
2734                                                              ;BIT15=1=>ERROR OCCURRED
2735                                                              ;BIT07=1=>DONE
2736                                                              ;BIT14-BIT09 AND BIT06-BIT03
2737                                                              ;INDICATE TYPE OF ERROR
2738  004164     000               DTADPB  BYTE   0              ;(0) DRIVE NUMBER
2739  004165     000                       .BYTE   0              ;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
2740  004166     000                       .BYTE   0              ;(2) COMMAND
2741  004167     000                       .BYTE   0              ;(3) PSEL AND A17 AND A16
2742  004170  000000                       .WORD   0              ;(4) WORD COUNT (MUST BE NEG )
2743  004172  047712                        WORD   BUFFER         ;(6) BUFFER ADDRESS OR
2744                                                              ;REGISTER TABLE POINTER
2745  004174     000                        BYTE   0              ;(10) SECTOR ADDRESS OR
2746                                                              ;FIRST REG. INDEX
2747  004175     000                        BYTE   0              ;(11) TRACK ADDRESS OR
2748                                                              ;LAST REG. INDEX
2749  004176  000000                        WORD   0              ,(12) CYLINDER ADDRESS
2750  004200  004204                        WORD   RP REG         ;(14) ERROR TABLE POINTER
2751                                                              ;POINTS TO THE FIRST OF TWENTY
2752                                                              ;LOCATIONS OF WHERE THE DRIVER
2753                                                              ;IS TO STORE THE RH11/RP04
2754                                                              ;REGISTERS ON AN ERROR. IF LEFT
2755                                                              ;ZERO REGISTERS ARE NOT SAVED.
2756  004202  000000                        WORD   0              ;(16) STATUS/ERROR INDICATOR
2757                                                              ;BIT15=1=>ERROR OCCURRED
2758                                                              ;BIT07=1=>DONE
2759                                                              ;BIT14-BIT09 AND BIT06-BIT03
2760                                                              ,INDICATE TYPE OF ERROR
2761
2762
2763
2764                                ,SAVE RH11/RP04 REGISTERS HERE ON ERROR
2765
2766  004204  000000                RP.REG:  WORD   0              ;RPCS1 (776700) CONTROL & STATUS #1
2767  004206  000000                        .WORD   0              ;RPWC (776702) WORD COUNT
2768  004210  000000                        .WORD   0              ;RPBA (776704) BUS ADDRESS
2769  004212  000000                        .WORD   0              ,RPDA (776706) DESIRED SECTOR/TRACK
2770  004214  000000                        .WORD   0              ;RPCS2 (776710) CONTROL & STATUS #2
2771  004216  000000                        .WORD   0              ;RPDS1 (776712) DISK STATUS
2772  004220  000000                        .WORD   0              ;RPER1 (776714) ERROR REG. #1
2773  004222  000000                        .WORD   0              ;RPAS (776716) ATTENTION SUMMARY
2774  004224  000000                        .WORD   0              ;RPLA (776720) LOOK AHEAD
2775  004226  000000                         WORD   0              ;RPDB (776722) DATA BUFFER
2776  004230  000000                        .WORD   0              ;RPMR (776724) MAINTAINABILITY
2777  004232  000000                        .WORD   0              ;RPDT (776726) DRIVE TYPE
2778  004234  000000                        .WORD   0              ;RPSN (776730) SERIAL NUMBER
2779  004236  000000                        .WORD   0              ;RPOF (776732) OFFSET
2780  004240  000000                        .WORD   0              ;RPCA (776734) DESIRED CYLINDER
2781  004242  000000                        .WORD   0              ;RPCC (776736) CURRENT CYLINDER
2782  004244  000000                         WORD   0              ;RPER2 (776740) ERROR REG #2
2783  004246  000000                         WORD   0              ,RPER3 (776742) ERROR REG #3
2784  004250  000000                        .WORD   0              ;RPEC1 (776744) ECC POSITION
2785  004252  000000                        .WORD   0              ,RPEC2 (776746) ECC PATTERN
2786
```

```
2787                                    ;STATUS/ERROR MESSAGE POINTER TABLE
2788  004254  044722                    STATBL: .WORD   MSGB14          ;OFFLINE OR UNSAFE DRIVE REQUESTED
2789  004256  044764                            .WORD   MSGB13          ;UNLOAD DRIVE REQUESTED
2790  004260  045015                            .WORD   MSGB12          ;PERSISTENT UNSAFE
2791  004262  045037                            .WORD   MSGB11          ;PARITY ERROR OCCURRED
2792  004264  045065                            .WORD   MSGB10          ;FATAL PARITY ERROR
2793  004266  045110                            .WORD   MSGB09          ;SOFTWARE TIMEOUT ON THIS DRIVE
2794  004270  045147                            .WORD   MSGB08          ;SOFTWARE TIMEOUT ON ANOTHER DRIVE
2795  004272  045211                             WORD   MSGB06          ;ERROR OCCURRED DURING I/O OPERATION
2796  004274  045255                            .WORD   MSGB05          ;ERROR OCCURRED DURING NON-I/O OPERATION
2797  004276  045325                             WORD   MSGB04          ;UNSAFE OCCURRED
2798  004300  045345                             WORD   MSGB03          ;AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
2799  004302  045415                            .WORD   MSGB02          ;DRIVE HAS NOT RESPONDED TO PORT REQUEST
2800  004304  045465                             WORD   MSGB01          ;DRIVE HAS BECOME NONEXISTENT
```

```
                                .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 $ITEMB  THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT
                                ,*NOTE1·          IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*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                          $ERRTB
                                ;*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
                                ,*    $ERRPC  $REG3

004306  044043                          EM1
004310  045523                          DH1
004312  047104                          DT1
004314  047536                          DF1

                                ,*  ERROR ITEM 2
                                ,*    UNEXPECTED ATTENTION OCCURRED
                                ,*    ERR PC  DRIVE   RPAS    RPDS1   RPER1     RPER2     RPER3
                                ,*    $ERRPC  $REG1   $REG3   RPERRS  RPERRS+2  RPERRS+4  RPERRS+6

004316  044106                          EM2
004320  045540                          DH2
004322  047110                          DT2
004324  047542                          DF2

                                ,*  ERROR ITEM 3
                                ,*    MASSBUS PARITY ERROR (MCPE=1)
                                ,*    TEST    ERR PC  ADDRESS DATA
                                ,*    $TMP0   $ERRPC  RD ADR  RD WRD

004326  044144                          EM3
004330  045626                          DH3
004332  047126                          DT3
004334  047546                          DF3

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

```
                                ;*    $TMPO   $ERRPC  WRT ADR WRT.WD  RD.WRD

004336  044201                        EM4
004340  045663                        DH4
004342  047136                        DT4
004344  047552                        DF4

                                ;*  ERROR ITEM 5
                                ;*    ADDRESS PLUG CHANGE BIT SET
                                ;*    ERR PC  DRIVE   RPAS    RPDS1   RPER1     RPER2    RPER3
                                ;*    $ERRPC  $REG1   $REG3   RPERRS  RPERRS+2  RPERRS+4 RPERRS+6

004346  044235                        EM5
004350  045540                        DH2
004352  047110                        DT2
004354  047542                        DF2

                                ;*  ERROR ITEM 6 -- NOT USED

004356  000000                        0
004360  000000                        0
004362  000000                        0
004364  000000                        0

                                ;*  ERROR ITEM 7 -- NOT USED

004366  000000                        0
004370  000000                        0
004372  000000                        0
004374  000000                        0

                                ;*  ERROR ITEM 10
                                ;*    RH11/RP04/5/6 FAILED TO RESPOND TO ADDRESSING
                                ;*    RPCS1   ERR PC
                                ;*    RH.ADR  $ERRPC

004376  044271                        EM10
004400  045732                        DH10
004402  047170                        DT10
004404  047556                        DF10

                                ;*  ERROR ITEM 11
                                ;*    DRIVE SELECTED IS NOT ONLINE
                                ;*    DRIVE   ERR PC
                                ;*    $REG2   $ERRPC

004406  044347                        EM11
004410  045751                        DH11
004412  047174                        DT11
004414  047562                        DF11

                                ;*  ERROR ITEM 12
                                ;*    IMPROPER HEADER DATA
                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
                                ;*    $TMPO   $ERRPC  $REG0   CHKDRV  CYL DS  TRK DS  SEC DS
                                ;*        GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR
```

```
                                ;*        CYL.DS  TRK DS  SEC.DS  CYL.RD  TRK RD  SEC RD
                                ;*    CYLNDR, TRACK, AND SECTOR ARE DECIMAL

004416  044404                          EM12
004420  045770                          DH12
004422  047200                          DT12
004424  047566                          DF12

                                ;*  ERROR ITEM 13
                                ;*    DATA COMPARE FAILURE
                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
                                ;*    $TMPO   $ERRPC  $REGO   CHKDRV  CYL DS  TRK DS  SEC DS
                                ;*        GDDAT   BDDAT   WRDCNT  GDADR   BDADR
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
                                ;*      CYLNDR, TRACK, SECTOR, AND WRDCNT  ARE DECIMAL

004426  044431                          EM13
004430  045770                          DH12
004432  047232                          DT13
004434  047576                          DF13

                                ;*  ERROR ITEM 14 -- FOLLOWS #13
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR

004436  000000                          0
004440  000000                          0
004442  047250                          DT13A
004444  047606                          DF14

                                ;*  ERROR ITEM 15
                                ;*    DATA COMPARE FAILURE
                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
                                ;*    $TMPO   $ERRPC  $REGO   CHKDRV  CYL DS  TRK DS  SEC DS
                                ;*        GDDAT   BDDAT   WRDCNT  GDADR   BDADR
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
                                ;*        CYLNDR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL

004446  044431                          EM13
004450  045770                          DH12
004452  047232                          DT13
004454  047576                          DF13

                                ;*  ERROR ITEM 16 -- FOLLOWS #15
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR

004456  000000                          0
004460  000000                          0
004462  047250                          DT13A
004464  047606                          DF14

                                ;*  ERROR ITEM 17
                                ;*    DISK ERROR IN TIMING TEST
                                ;*    TEST    ERR PC  DRIVE   RPCS1   RPDS1     RPER1     RPER2     RPER3
                                ;*    $TMPO   $ERRPC  CHKDRV  RP REG  RP REG+12 RP REG+14 RP REG+40 RP REG+42

004466  044456                          EM17
```

```
2969  004470  046204                    DH17
2970  004472  047262                    DT17
2971  004474  047612                    DF17
2972
2973                                ,*  ERROR ITEM 20
2974                                ,*    CLOCK (KW11-P) OVERFLOW IN TIMING TEST
2975                                ,*    TEST    ERR PC  DRIVE   RPCS1   RPDS1     RPER1     RPER2     RPER3
2976                                ;*    $TMPO   $ERRPC  CHKDRV  RP.REG  RP.REG+12 RP REG+14 RP REG+40 RP REG+42
2977
2978  004476  044510                    EM20
2979  004500  046204                    DH17
2980  004502  047262                    DT17
2981  004504  047612                    DF17
2982
2983                                ,*  ERROR ITEM 21
2984                                ;*    DATA COMPARE FAILURE
2985                                ,*    TEST      ERR PC  TST PC  DRIVE   CYLNDR  TRACK
2986                                ,*    $TMPO     $ERRPC  $REGO   CHKDRV  CYL DS  TRK DS
2987                                ,*        GDDAT   BDDAT   WRDCNT  SECTOR
2988                                ,*        $REG1   $BDDAT  $REG4   $REG1
2989                                ,*        CYLINDR, TRACK, WRDCNT, AND SECTOR ARE DECIMAL
2990
2991  004506  044431                    EM13
2992  004510  046302                    DH21
2993  004512  047302                    DT21
2994  004514  047616                    DF21
2995
2996                                ,*  ERROR ITEM 22--FOLLOWS #21
2997                                ,*        $REG1   $BDDAT   $REG4   $PEG1
2998
2999  004516  000000                      0
3000  004520  000000                      0
3001  004522  047316                      DT21A
3002  004524  047626                      DF22
3003
3004                                ,*  ERROR ITEM 23
3005                                ,*      DISK ERROR DURING SEEK
3006                                ,*      TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2     RPDS1
3007                                ,*      $TMPO   $ERRPC  CHKDRV  CYL DS  RP REG RP REG+10 RP REG+12
3008                                ,*          RPER1     RPER2     RPER3     RPCA      RPCC
3009                                ,*          RP REG+14 RP REG+40 RP REG+42 RP REG+34 RP REG+36
3010
3011  004526  044557                      EM23
3012  004530  046417                      DH23
3013  004532  047326                      DT23
3014  004534  047632                      DF23
3015
3016                                ,*  ERROR ITEM 24
3017                                ,*      SEEK NOT COMPLETE WITHIN 120 MS
3018                                ,*      TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2     RPDS1
3019                                ,*      $TMPO   $ERRPC  CHKDRV  CYL DS  RP REG RP REG+10 RP REG+12
3020                                ,*          RPER1     RPER2     RPER3     RPCA      RPCC
3021                                ,*          RP REG+14 RP REG+40 RP REG+42 RP REG+34 RP REG+36
3022
3023  004536  044606                      EM24
3024  004540  046417                      DH23
```

```
                                        DT23
                                        DF23


                                ,,**********************************************************************
                                ,,**********************************************************************
                                ,*  ERROR ITEMS 23-40 NOT USED
                                ,*  ERROR ITEMS 41-46 WILL HAVE AN EM THAT
                                ,*  VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM
                                ,*  RH11/RP04/5/6 ERROR (MESSAGE)
                                ,*  WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING·
                                ,*  1)  OFFLINE OR UNSAFE DRIVE REQUESTED
                                ,*  2)  UNLOADED DRIVE REQUESTED
                                ,*  3)  PERSISTENT UNSAFE
                                ,*  4)  PARITY ERROR OCCURRED
                                ,*  5)  FATAL PARITY ERROR
                                ,*  6)  SOFTWARE TIMEOUT ON THIS DRIVE
                                ,*  7)  SOFTWARE TIMEOUT ON ANOTHER DRIVE
                                ,*  8)  ERROR OCCURRED DURING I/O OPERATION
                                ,*  9)  ERROR OCCURRED DURING NON-I/O OPERATION
                                ,*  10) UNSAFE OCCURRED
                                ,*  11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED

004546                          ITEM41

                                ,*  ERROR ITEM 41
                                ,*    RH11/RP04/5/6 ERROR (MESSAGE)
                                ,*    TEST    ERR PC  TST PC  DRIVE
                                ,*    $TMP0   $ERRPC  $REG0   CHKDRV

004546  044646                    EM41
004550  046552                    DH41
004552  047356                    DT41
004554  047642                    DF41

                                ,*  ERROR ITEM 42
                                ,*    RH11/RP04/5/6 ERROR (MESSAGE)
                                ,*    TEST    ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1
                                ,*    $TMP0   $ERRPC  $REG0   CHKDRV  RP REG  RP REG+10 RP REG+12

004556  044646                    EM41
004560  046610                    DH42
004562  047366                    DT42
004564  047646                    DF42

                                ,*  ERROR ITEM 43
                                ,*    RH11/RP04/5/6 ERROR (MESSAGE)
                                ,*    TEST    ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1
                                ,*    $TMP0   $ERRPC  $REG0   CHKDRV  RP REG  RP REG+10 RP REG+12
                                ,*        RPER1     RPER2     RPER3
                                ,*        RP REG+14 RP REG+40 RP REG+42

004566  044646                    EM41
004570  046610                    DH42
004572  047404                    DT43
```

```
3081  004574  047652                          DF43
3082
3083                                  ;*  ERROR ITEM 44
3084                                  ;*    RH11/RP04/5/6 ERROR (MESSAGE)
3085                                  ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
3086                                  ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL DS  TRK DS    SEC DS
3087                                  ;*        RPCS1     RPCS2     RPDS1     RPCC    RPCA     RPDA
3088                                  ;*        RP REG    RP.REG+10 RP.REG+12 RP.REG+36 RP REG+34 RP REG+06
3089                                  ;*      RPER1       RPER2       RPER3
3090                                  ;*      RP REG+14   RP.REG+40   RP REG+42
3091                                  ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
3092
3093  004576  044646                          EM41
3094  004600  045770                          DH12
3095  004602  047430                          DT44
3096  004604  047662                          DF44
3097
3098                                  ;*  ERROR ITEM 45
3099                                  ;*    RH11/RP04/5/6 ERROR (MESSAGE)
3100                                  ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
3101                                  ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL DS  TRK DS    SEC DS
3102                                  ;*        RPCS1     RPCS2     RPDS1     RPCC      RPCA      RPDA
3103                                  ;*        RP REG    RP REG+10 RP REG+12 RP REG+36 RP REG+34 RP REG+06
3104                                  ;*      RPER1       RPER2       RPER3       RPWC      RPBA     RPDB
3105                                  ;*      RP REG+14   RP REG+40   RP REG+42   RP REG+2   RP REG+4 RP REG+22
3106                                  ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
3107
3108  004606  044646                          EM41
3109  004610  045770                          DH12
3110  004612  047470                          DT45
3111  004614  047676                          DF45
3112
3113                                ;*  ERROR ITEM 46
3114                                ;*    FATAL WRITE CHECK ERROR (MESSAGE)
3115                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
3116                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL DS  TRK DS    SEC DS
3117                                ;*        RPCS1     RPCS2     RPDS1     RPCC      RPCA      RPDA
3118                                ;*        RP REG    RP REG+10 RP REG+12 RP REG+36 RP REG+34 RP REG+06
3119                                ;*      RPER1       RPER2       RPER3       RPWC      RPBA     RPDB
3120                                ;*      RP REG+14   RP REG+40   RP REG+42   RP REG+2   RP REG+4 RP REG+22
3121                                ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
3122
3123  004616  044672                          EM46
3124  004620  045770                          DH12
3125  004622  047470                          DT45
3126  004624  047676                          DF45
3127
```

```
3128
3129                                    SBTTL  START OF PROGRAM
3130
3131
3132  004626  012737  177777  001226  START3  MOV     #-1,@#BUSADR    ,GET BUSADR FLAG
3133  004634  000402                          BR      STRT1A
3134  004636  005037  001226          START1: CLR     @#BUSADR        ,CLR BUSADR FLAG
3135  004642  005037  001224          STRT1A: CLR     @#CNTRLC        ,NO CONTROL "C"
3136  004646  000411                          BR      START
3137  004650  012737  177777  001226  START4  MOV     #-1,@#BUSADR    ,SET BUSADR FLAG
3138  004656  000402                          BR      STRT2A
3139  004660  005037  001226          START2  CLR     @#BUSADR        ,CLR BUSADR FLAG
3140  004664  012737  177777  001224  STRT2A  MOV     #-1,@#CNTRLC    ,SET CONTROL "C" FLAG
3141  004672  000005                  START:  RESET
3142                                    SBTTL  INITIALIZE THE COMMON TAGS
3143                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3144  004674  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3145  004700  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
3146  004702  022706  001140                  CMP     #SWR,R6 ;;DONE?
3147  004706  001374                          BNE     -6              ;;LOOP BACK IF NO
3148  004710  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
3149                                  ;;INITIALIZE A FEW VECTORS
3150  004714  012737  022606  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3151  004722  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
3152  004730  012737  017634  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3153  004736  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
3154  004744  012737  023134  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3155  004752  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
3156  004760  012737  176543  023606          MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
3157  004766  012737  123456  023610          MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
3158  004774  005037  001204                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
3159  005000  005037  001206                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3160  005004  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
3161  005012  012737  005012  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3162  005020  012737  005020  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
3163                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER.  IF NOT FOUND OR IT IS
3164                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER
3165  005026  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
3166  005032  012737  005066  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
3167  005040  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
3168  005046  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
3169  005054  022777  177777  174056          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
3170  005062  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3171                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
3172  005064  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
3173  005066  012716  005074          64$     MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
3174  005072  000002                          RTI
3175  005074  012737  000176  001140  65$     MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
3176  005102  012737  000174  001142          MOV     #DISPREG,DISPLAY
3177  005110  012637  000004          66$     MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
3178
3179  005114  012700  001160                  MOV     #$REGAD,R0      ;FIRST ADDRESS
3180  005120  005020                  1$      CLR     (R0)+           ;CLEAR VARIABLE STORAGE
3181  005122  022700  001210                  CMP     #$BELL,R0       ;DONE?
3182  005126  001374                          BNE     1$              ;NO--BRANCH
3183  005130  013737  001414  001150          MOV     @#TPS,@#$TPS    ;SETUP THE STATUS AND BUFFER REG'S
```

```
3184  005136  013737  001416  001152          MOV     @#TPB,@#$TPB    ;FOR THE TYPE ROUTINE
3185  005144  005227  177777                  INC     #-1             ;FIRST START ?
3186  005150  001032                          BNE     3$              ;BR IF NOT
3187  005152  023737  000042  000046          CMP     @#42,@#46       ;ACT11 AUTOMATIC MODE?
3188  005160  001002                          BNE     4$              ;YES, SKIP TITLE PRINTOUT
3189  005162  104401  047712                  TYPE    ,TITLE          ;TYPE THE PROGRAM'S TITLE
3190  005166  005737  000042          4$      TST     42              ;AUTO ACCEPT OR CHAIN MODE ?
3191  005172  001006                          BNE     2$              ;BR IF EITHER
3192  005174  122737  000011  000041          CMPB    #11,41          ;LOADED FROM AN RP04/5/6 ?
3193  005202  001002                          BNE     2$              ;BR IF NOT
3194  005204  104401  050012                  TYPE    ,LOADRV         ;INSTRUCT THE OPERATOR TO REMOVE THE PACK
3195                                                                  ;ON DRIVE 0 IF DRIVE 0 IS TO BE TESTED
3196  005210  105737  000041          2$      TSTB    @#41            ;LOADED FROM PAPER TAPE ?
3197  005214  001410                          BEQ     3$              ;BR IF NOT
3198  005216  004737  050306                  JSR     PC,$SIZE        ;SIZE THE MEMORY
3199  005222  023727  050402  100000          CMP     $LSTAD,#100000  ;16K OR MORE ON THE SYSTEM ?
3200  005230  103002                          BHIS    3$              ;BR IF YES
3201  005232  104401  050210                  TYPE    ,NOLOAD         ;INFORM THE OPEATOR THAT THE 'XXDP' LOADER
3202                                                                  ;WILL BE OVERWRITTEN
3203  005236  004737  021344          3$      JSR     PC,$TKINT       ;TURN ON THE TTY KEYBOARD INTERRUPT
3204                                   SBTTL   GET VALUE FOR SOFTWARE SWITCH REGISTER
3205  005242  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
3206  005246  001006                          BNE     67$             ;;BRANCH IF YES
3207  005250  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
3208  005256  001005                          BNE     68$             ;;BRANCH IF NO
3209  005260  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
3210  005262  000403                          BR      68$
3211  005264  112737  000001  001134  67$     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
3212  005272                          68$
3213  005272  005227  177777                  INC     #-1             ;SEE IF FIRST START
3214  005276  001002                          BNE     SRTINT          ;BR IF NOT
3215  005300  004737  050404                  JSR     PC,GETADR       ;GET OR CHECK THE RH11 ADDRESS
3216  005304  104401  001215          SRTINT  TYPE    ,$CRLF          ;CR-LF
3217  005310  004737  024032                  JSR     PC,@#LP AVL     ;CHECK FOR A LINE PRINTER
3218  005314  005037  177776                  CLR     @#PS            ;ENSURE THE PRIORITY = 0
3219  005320  012737  000001  001104          MOV     #1,$ICNT        ;SET ITERATION COUNT TO 1
3220  005326  004737  031424                  JSR     PC,@#GETSWR     ;GO CHECK FOR CONTROL SWITCHES
3221  005332  004737  024074                  JSR     PC,@#ST CLK     ;INITIALIZE THE CLOCK
3222  005336  004737  034532          SETVEC  JSR     PC,RPINIT       ;CHECK THE DRIVE STATUS
3223  005342  012737  177777  034454          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
3224  005350  062727  177777  000000          ADD     #-1,#0          ;FIRST START ?
3225  005356  103003                          BCC     11$             ;BR IF YES
3226  005360  005737  001224                  TST     CNTRLC          ;CONTROL 'C' SWITCH SET ?
3227  005364  001102                          BNE     SRTDRV          ;CONTINUE IF YES
3228  005366  012737  000340  177776  11$     MOV     #PR7,PS         ;SET PRIORITY TO 7
3229  005374  005004                          CLR     R4              ;DRIVE TABLE POINTER
3230  005376  104401  001215                  TYPE    ,$CRLF          ;CR-LF
3231  005402  104401  043100                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
3232  005406                          1$
3233  005406  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
3234                                                                  ;;TYPE DRIVE NUMBER
3235  005410  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3236  005412     002                           BYTE    2               ;;TYPE 2 DIGIT(S)
3237  005413     000                           BYTE    0               ;;SUPPRESS LEADING ZEROS
3238  005414  104401  044040                  TYPE    ,MSG SP         ;SPACES
3239  005420  104401  044040                  TYPE    ,MSG SP         ;SPACES
```

```
3240  005424  105764  034366              TSTB    DRVSTA(R4)          ;CHECK DRIVE'S STATUS
3241  005430  100416                      BMI     4$                  ;BR IF UNSAFE
3242  005432  001020                      BNE     5$                  ;BR IF ONLINE
3243  005434  105764  034376              TSTB    DRVTYP(R4)          ;SEE IF OFFLINE OR NONEXISTENT
3244  005440  001404                      BEQ     2$                  ,BR IF NONEXISTENT
3245  005442  100006                      BPL     3$                  ,BR IF OFFLINE
3246  005444  104401  043174              TYPE    ,NOTRP              ;DRIVE NOT AN RP04/5/6
3247  005450  000440                      BR      9$                  ;CHECK NEXT DRIVE
3248  005452  104401  043147        2$    TYPE    ,NOTPRS             ;DRIVE NOT PRESENT
3249  005456  000435                      BR      9$                  ;CHECK NEXT DRIVE
3250  005460  104401  043126        3$    TYPE    ,UNTOFF             ;DRIVE OFFLINE
3251  005464  000405                      BR      6$                  ;PRINT DRIVE TYPE
3252  005466  104401  043164        4$    TYPE    ,NOTSAF             ,DRIVE UNSAFE
3253  005472  000402                      BR      6$                  ,PRINT DRIVE TYPE
3254  005474  104401  043137        5$.   TYPE    ,UNTON              ;DRIVE ONLINE
3255  005500  104401  044040        6$.   TYPE    ,MSG.SP             ;SPACES
3256  005504  012737  043212  005550      MOV     #RP04B,8$           ,ADDRESS OF RP04 MESSAGE
3257  005512  132764  000001  034376      BITB    #BIT00,DRVTYP(R4)   ;RP04 ?
3258  005520  001012                      BNE     7$                  ,BR IF YES
3259  005522  012737  043217  005550      MOV     #RP05,8$            ,ADDRESS OF RP05 MESSAGE
3260  005530  132764  000002  034376      BITB    #BIT01,DRVTYP(R4)   ;RP05 ?
3261  005536  001003                      BNE     7$                  ;BR IF YES
3262  005540  012737  043224  005550      MOV     #RP06,8$            ,ADDRESS OF RP06 MESSAGE
3263  005546  104401              7$      TYPE                        ;TYPE THE DRIVE TYPE MESSAGE
3264  005550  000000              8$       WORD   0                   ;MESSAGE ADDRESS HERE
3265  005552  104401  001215        9$    TYPE    ,$CRLF              ,CR-LF
3266  005556  005204                      INC     R4                  ;INCREMENT DRIVE NUMBER/TABLE POINTER
3267  005560  020427  000010              CMP     R4,#8               ,FINISHED ?
3268  005564  001310                      BNE     1$                  ,BR IF NOT
3269  005566  005037  177776              CLR     PS                  ,SET PRIORITY BACK TO '0'
3270  005572  005737  001224        SRTDRV TST    @#CNTRLC            ;CONTROL "C" START/RESTART?
3271  005576  001417                      BEQ     1$                  ;NO--BRANCH
3272  005600  013746  001222              MOV     SAVCSW,-(SP)        ,GET THE PREVIOUS 'C SWR' CONTENTS
3273  005604  063716  001220              ADD     C SWR,(SP)          ,SET UP TO SEE IF 'BIT00' IS DIFFERENT
3274  005610  032726  000001              BIT     #BIT00,(SP)+        ;IS 'BIT00' DIFFERENT ?
3275  005614  001405                      BEQ     9$                  ,BR IF NOT
3276  005616  013737  001220  001222      MOV     C SWR,SAVCSW        ,STORE PRESENT 'C SWR' VALUE
3277  005624  004737  024352              JSR     PC,LODFLT           ,RESET PARAMETERS TO THEIR DEFAULT VALUES
3278  005630  004737  031654        9$    JSR     PC,@#GT PRM         ,GET PARAMETERS
3279  005634  000420                      BR      4$
3280  005636  004737  024352        1$    JSR     PC,LODFLT           ,SETUP DEFAULT PARAMETERS
3281  005642  005037  001232              CLR     DRVSEL              ,NO DRIVES SELECTED
3282  005646  005000                      CLR     R0                  ,DETERMINE THE DRIVES THAT
3283  005650  012701  000001              MOV     #1,R1               ,ARE AVAILABLE FOR TESTING
3284  005654  105760  034366        2$    TSTB    DRVSTA(R0)
3285  005660  003403                      BLE     3$
3286  005662  156037  034502  001232      BISB    ATABIT(R0),@#DRVSEL
3287  005670  005200              3$      INC     R0
3288  005672  106301                      ASLB    R1
3289  005674  001367                      BNE     2$
3290  005676  005037  034456        4$    CLR     @#SEEKFG            ,CLEAR SEEK FLAG
3291  005702  032737  000400  001220      BIT     #SW08,@#C SWR       ,DO SEEK BEFORE DATA TRANSFER?
3292  005710  001002                      BNE     5$                  ,YES--BRANCH
3293  005712  005137  034456              COM     @#SEEKFG            ,NO
3294  005716  122737  000011  000041  5$  CMPB    #11,41   ,LOADED FROM AN RP04/5/6 ?
3295  005724  001003                      BNE     10$                 ,BR IF NOT
```

```
3296  005726  042737  000001  001232          BIC     #BIT00,DRVSEL   ;CLEAR THE DRIVE 0 SELECTION BIT
3297  005734  104401  043231          10$     TYPE    ,DRIVES         ;'DRIVES(S) TO BE TESTED'
3298  005740  005037  017550                  CLR     @#$ENDCT        ;DETERMINE PASSES TO MAKE AND
3299  005744  005000                          CLR     R0              ;THE DRIVES TO BE TESTED
3300  005746  013701  001232                  MOV     @#DRVSEL,R1     ;ANY DRIVES SELECTED?
3301  005752  001004                          BNE     6$              ;YES--BRANCH
3302  005754  104401  043262                  TYPE    ,NONE           ;'NONE'
3303
3304  005760  000137  017366                  JMP     @#$EOP          ;GO TO END OF PROGRAM
3305  005764  006201                  6$.     ASR     R1              ;REPORT THE DRIVES TO BE TESTED
3306  005766  103011                          BCC     7$
3307  005770  005237  017550                  INC     @#$ENDCT        ;GIVE THIS DRIVE A PASS
3308  005774  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
3309  005776  104403                          TYPOS                   ,;GO TYPE--OCTAL ASCII
3310  006000     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3311  006001     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3312  006002  005701                          TST     R1              ,MORE DRIVES?
3313  006004  001404                          BEQ     8$              ;NO--BRANCH
3314  006006  104401  043267                  TYPE    ,COMMA          ,','
3315  006012  005200                  7$      INC     R0              ;FORM DRIVE NUMBER
3316  006014  000763                          BR      6$
3317  006016  013737  017550  017542  8$      MOV     @#$ENDCT,@#$EOPCT
3318  006024  005737  001244                  TST     @#CLKSTA        ,KW11-P AVAILABLE
3319  006030  003006                          BGT     RSTRT1          ;YES--BRANCH
3320  006032  032737  036000  001234          BIT     #36000,@#TSTNMS ;NO--ANY TIMING TESTS TO BE PERFORMED?
3321  006040  001402                          BEQ     RSTRT1          ;NO--BRANCH
3322  006042  104401  043271                  TYPE    ,NOCLOK         ;'NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED'
3323  006046  005737  001232          RSTRT1  TST     DRVSEL          ,ANY DRIVES SELECTED ?
3324  006052  001002                          BNE     1$              ;BR IF YES
3325  006054  000137  004660                  JMP     START2          ,GET DRIVE SELECTION ENTRY
3326  006060  005037  001254          1$      CLR     CHKDRV          ,INIT.  THE CHECK DRIVE KEY
3327  006064  012737  000001  001256          MOV     #1,DRVMSK       ;START TO CHECK DESIRED DRIVES
3328  006072  033737  001256  001232  RSTRT2  BIT     DRVMSK,DRVSEL   ,IS THIS DRIVE SELECTED?
3329  006100  001010                          BNE     DRVOK           ,YES--GO CHECK IF DRIVE IS READY FOR TESTING
3330  006102  012706  001100          RESTART MOV     #STACK,SP       ;SETUP THE STACK POINTER
3331  006106  005237  001254                  INC     CHKDRV          ,MOVE  TO NEXT DRIVE NUMBER
3332  006112  106337  001256                  ASLB    DRVMSK          ;POSITION THE MASK
3333  006116  103753                          BCS     RSTRT1          ,BRANCH IF THE DRIVE NUMBER NEEDS INITIALIZED
3334  006120  000764                          BR      RSTRT2
3335
3336  006122  013702  001254          DRVOK   MOV     CHKDRV,R2       ,PICKUP THE DRIVE NUMBER
3337  006126  105762  034366                  TSTB    DRVSTA(R2)      ,IS DESIRED DRIVE ON-LINE?
3338  006132  003005                          BGT     1$              ,YES, BRANCH
3339  006134  104011                          ERROR   11              ,DRIVE SELECTED IS NOT ONLINE
3340  006136  043737  001256  001232          BIC     DRVMSK,DRVSEL   ,CLEAR DRIVE'S SELECTION BIT
3341  006144  000756                          BR      RESTART         ;RETURN
3342  006146  010237  004104          1$      MOV     R2,@#DPB A      ,SET THE DRIVE NUMBER INTO THE DPB'S
3343  006152  010237  004124                  MOV     R2,@#DPB.B
3344  006156  010237  004144                  MOV     R2,@#DPB.C
3345  006162  010237  004164                  MOV     R2,@#DTADPB
3346  006166  004737  024766                  JSR     PC,@#LDCMD      ,LOAD COMMAND INTO DPB B AND DPB C
3347  006172  012737  017366  001252          MOV     #$EOP,@#BYPASS  ,IF ERROR GO TO END OF PROGRAM
3348  006200  112737  000020  004105          MOVB    #FMT22/256 ,DPB A+1  ;ASSUME 16 BIT FORMAT
3349  006206  032737  000001  001220          BIT     #BIT00,C.SWR    ,16 BIT FORMAT REQUESTED ?
3350  006214  001402                          BEQ     2$              ;BR IF YES
3351  006216  105037  004105                  CLRB    DPB A+1         ,CLEAR THE 'FMT22' BIT
```

```
3352  006222  112737  000143  004106  2$:     MOVB    #SETFORM,DPB.A+2  ;SET THE FORMAT BIT PER DPB.A+1
3353  006230  004037  025032                  JSR     R0,@#CALL A       ;GO EXECUTE THE COMMAND
3354  006234  112737  000107  004106          MOVB    #RECAL,@#DPB.A+2          ;RECAL=COMMAND
3355  006242  004037  025032                  JSR     R0,@#CALL.A       ;GO EXECUTE THE COMMAND
3356  006246  104401  043357                  TYPE    ,TESTNG           ;'TESTING DRIVE '
3357  006252  010246                          MOV     R2,-(SP)          ;;SAVE R2 FOR TYPEOUT
3358  006254  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
3359  006256     001                          .BYTE   1                 ;;TYPE 1 DIGIT(S)
3360  006257     000                           BYTE   0                 ;;SUPPRESS LEADING ZEROS
3361  006260  104401  044040                  TYPE    ,MSG SP           ;TYPE SPACES
3362  006264  104401  043401                  TYPE    ,SERIAL           ;'SERIAL NUMBER '
3363  006270  012700  000004                  MOV     #4,R0             ;FOUR DIGITS TO TYPE
3364  006274  013701  004234                  MOV     RP REG+30,R1      ;SERIAL NUMBER
3365  006300  005002                  3$      CLR     R2                ,ZERO
3366  006302  006101                          ROL     R1                ,PUT THE NEXT DIGIT
3367  006304  006102                          ROL     R2                ,INTO R2
3368  006306  006101                          ROL     R1
3369  006310  006102                          ROL     R2
3370  006312  006101                          ROL     R1
3371  006314  006102                          ROL     R2
3372  006316  006101                          ROL     R1
3373  006320  006102                          ROL     R2
3374  006322  062702  000060                  ADD     #'0,R2            ,MAKE IT ASCII
3375  006326  010227                          MOV     R2,(PC)+          ,SAVE IT
3376  006330  000000                  4$       WORD    0
3377  006332  104401  006330                  TYPE    ,4$               ,TYPE
3378  006336  005300                          DEC     R0                ,ALL DIGITS TYPED?
3379  006340  003357                          BGT     3$                ,NO -- BRANCH
3380  006342  104401  001215                  TYPE    ,$CRLF
3381  006346  113737  001364  001115          MOVB    ERR CT,$ERMAX     ,SETUP MAX ERROR COUNT
```

```
                                    SBTTL  #### TESTS ####

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

        ;*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

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

                                    .SBTTL  *** SEEK TESTS ***

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

        ;*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.

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

                                    ;;*************************************************************
                                    ;*TEST 0        RECAL/SEEK TEST
```

```
3438
3439                                    ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
3440                                    ;*      COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC" AT
3441                                    ,*      THE COMPLETION OF BOTH COMMANDS STATUS INDICATIONS ARE
3442                                    ,*      CHECKED TO ENSURE NO ERRORS OCCURRED.
3443
3444                                    ;;**********************************************************************
3445  006354                            TST0:
3446  006354  000240                           NOP
3447  006356  033737  001424  001234           BIT     @#BITS+<0*2>,TSTNMS ;DO THIS TEST?
3448  006364  001002                           BNE     64$                 ;YES--BRANCH
3449  006366  000137  006516                   JMP     TST1                ;NO--GO TO THE NEXT TEST
3450  006372  012737  000000  001102  64$:     MOV     #0,@#$TSTNM         ;SET UP TEST NUMBER AND
3451                                                                        ;CLEAR THE ERROR FLAG ($ERFLG)
3452  006400  004737  024610                   JSR     PC,LODPRM           ;LOAD THE PARMETERS FOR THE TEST
3453  006404  012737  006500  001110           MOV     #TEST0,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3454  006412  013777  001102  172522           MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3455  006420  013737  001506  001204           MOV     @#RPT,$TIMES        ;GET THE ITERATION COUNT
3456  006426  112737  000031  001115           MOVB    #25.,$ERMAX         ;MAX ERRORS ALLOWED FOR TEST
3457  006434  112737  000107  004106           MOVB    #RECAL,@#DPB.A+2            ;RECAL=COMMAND
3458  006442  113737  001524  004134           MOVB    @#FS,@#DPB.B+10 ;FS
3459  006450  113737  001516  004135           MOVB    @#FT,@#DPB.B+11 ;FT
3460  006456  013737  001512  004136           MOV     @#LC,@#DPB.B+12 ;LC
3461  006464  012737  006514  001252           MOV     #EXIT0,@#BYPASS ;GO TO EXIT0 ON ERROR
3462  006472  012737  006500  001106           MOV     #TEST0,$LPADR   ;SETUP LOOP ADDRESS
3463  006500  012706  001100           TEST0   MOV     #STACK,SP       ;SET UP STACK POINTER
3464  006504  004037  025032                   JSR     R0,@#CALL.A     ;GO EXECUTE THE COMMAND
3465  006510  004037  025144                   JSR     R0,@#CALL.B     ,GO EXECUTE THE COMMAND
3466  006514  000004                   EXIT0.  SCOPE                   ;LOOP
3467
3468                                    ,,**********************************************************************
3469                                    ,*TEST 1        SEEK/SEEK TEST
3470
3471                                    ,*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
3472                                    ,*      CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
3473                                    ,*      "FC", "FT", "FS".  AT THE COMPLETION OF EACH SEEK, THE PROPER
3474                                    ,*      INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION
3475                                    ,*      "LC" WILL DEFAULT TO 128 AND "FC", "FT", "LT", "FS", AND "LS"
3476                                    ,*      WILL DEFAULT TO 0
3477
3478                                    ,,**********************************************************************
3479  006516                            TST1:
3480  006516  000240                           NOP
3481  006520  033737  001426  001234           BIT     @#BITS+<1*2>,TSTNMS ,DO THIS TEST?
3482  006526  001002                           BNE     64$                 ,YES--BRANCH
3483  006530  000137  006674                   JMP     TST2                ,NO--GO TO THE NEXT TEST
3484  006534  012737  000001  001102  64$     MOV     #1,@#$TSTNM         ,SET UP TEST NUMBER AND
3485                                                                        ,CLEAR THE ERROR FLAG ($ERFLG)
3486  006542  004737  024610                   JSR     PC,LODPRM           ,LOAD THE PARMETERS FOR THE TEST
3487  006546  012737  006656  001110           MOV     #TEST1,@#$LPERR  ,SETUP THE LOOP ON ERROR ADDRESS
3488  006554  013777  001102  172360           MOV     $TSTNM,@DISPLAY  ,LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3489  006562  013737  001506  001204           MOV     @#RPT,$TIMES        ,GET THE ITERATION COUNT
3490  006570  112737  000031  001115           MOVB    #25.,$ERMAX         ,MAX ERRORS ALLOWED FOR TEST
3491  006576  113737  001524  004134           MOVB    @#FS,@#DPB B+10 ,FS
3492  006604  113737  001526  004154           MOVB    @#LS,@#DPB C+10 ,LS
3493  006612  113737  001516  004135           MOVB    @#FT,@#DPB B+11 ,FT
```

```
3494  006620  113737  001520  004155          MOVB    @#LT,@#DPB.C+11 ;LT
3495  006626  013737  001510  004136          MOV     @#FC,@#DPB.B+12 ;FC
3496  006634  013737  001512  004156          MOV     @#LC,@#DPB.C+12 ;LC
3497  006642  012737  006672  001252          MOV     #EXIT1,@#BYPASS ;GO TO EXIT1 ON ERROR
3498  006650  012737  006656  001106          MOV     #TEST1,$LPADR   ;SETUP LOOP ADDRESS
3499  006656  012706  001100          TEST1:  MOV     #STACK,SP       ;SET THE STACK POINTER
3500  006662  004037  025334                  JSR     R0,@#CALL.C     ;GO EXECUTE THE COMMAND
3501  006666  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
3502  006672  000004                  EXIT1:  SCOPE                   ;LOOP
3503
3504                                  ;;**********************************************************************
3505                                  ;*TEST 2         INCREMENT/SEEK TEST
3506
3507                                  ;*      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
3508                                  ;*      CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC"
3509                                  ;*      WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS
3510                                  ;*      "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING
3511                                  ;*      AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC"
3512                                  ;*      UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH
3513                                  ;*      SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
3514                                  ;*      ENSURE PROPER OPERATION.
3515
3516                                  ;;**********************************************************************
3517  006674                          TST2
3518  006674  000240                          NOP
3519  006676  033737  001430  001234          BIT     @#BITS+(2*2),TSTNMS ,DO THIS TEST?
3520  006704  001002                          BNE     64$             ,YES--BRANCH
3521  006706  000137  007112                  JMP     TST3            ;NO--GO TO THE NEXT TEST
3522  006712  012737  000002  001102  64$     MOV     #2,@#$TSTNM     ,SET UP TEST NUMBER AND
3523                                                                  ,CLEAR THE ERROR FLAG ($ERFLG)
3524  006720  004737  024610                  JSR     PC,LOOPRM       ,LOAD THE PARMETERS FOR THE TEST
3525  006724  012737  007004  001110          MOV     #TEST2,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3526  006732  013777  001102  172202          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3527  006740  013737  001506  001204          MOV     @#RPT,$TIMES    ,GET THE ITERATION COUNT
3528  006746  112737  000031  001115          MOVB    #25.,$ERMAX     ,MAX ERRORS ALLOWED FOR TEST
3529  006754  012737  006762  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
3530  006762  113737  001524  004134  1$      MOVB    @#FS,@#DPB.B+10 ;FS
3531  006770  113737  001516  004135          MOVB    @#FT,@#DPB.B+11 ;FT
3532  006776  012737  007110  001252          MOV     #EXIT2,@#BYPASS ,GO TO EXIT2 ON ERROR
3533  007004  013737  001510  004136  TEST2   MOV     @#FC,@#DPB B+12 ;FC
3534  007012  012737  007012  001110          MOV     # ,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
3535  007020  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
3536  007024                          INCSK
3537  007024  004037  025144                  JSR     R0,@#CALL.B     ,GO EXECUTE THE COMMAND
3538  007030  063737  001514  004136          ADD     @#IC,@#DPB B+12 ;MOVE TO NEXT CYLINDER
3539  007036  023737  001512  004136          CMP     @#LC,@#DPB B+12 ;OUT OF CYLINDERS?
3540  007044  002367                          BGE     INCSK           ,NO--BRANCH
3541  007046  013737  001512  004136          MOV     @#LC,@#DPB.B+12
3542  007054  012737  007054  001110          MOV     # ,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
3543  007062  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
3544  007066                          DECSK
3545  007066  004037  025144                  JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
3546  007072  163737  001514  004136          SUB     @#IC,@#DPB B+12
3547  007100  023737  001510  004136          CMP     @#FC,@#DPB B+12
3548  007106  003767                          BLE     DECSK
3549  007110  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.

                                        ;;*****************************************************************
007112                                  TST3
007112  000240                                  NOP
007114  033737  001432  001234                  BIT     @#B!TS+<3*2>,TSTNMS ;DO THIS TEST?
007122  001002                                  BNE     64$             ;YES--BRANCH
007124  000137  007306                          JMP     TST4            ;NO--GO TO THE NEXT TEST
007130  012737  000003  001102  64$     MOV     #3,@#$TSTNM     ;SET UP TEST NUMBER AND
                                                                ;CLEAR THE ERROR FLAG ($ERFLG)
007136  004737  024610                          JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
007142  012737  007222  001110                  MOV     #TEST3,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
007150  013777  001102  171764                  MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
007156  013737  001506  001204                  MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
007164  112737  000031  001115                  MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
007172  012737  007200  001106                  MOV     #1$,$LPADR      ;SETUP TEST LOOP ADDRESS
007200  113737  001524  004134  1$      MOVB    @#FS,@#DPB.B+10 ;FS
007206  113737  001516  004135                  MOVB    @#FT,@#DPB.B+11 ;FT
007214  012737  007304  001252                  MOV     #EXIT3,@#BYPASS ;GO TO BYPASS ON ERROR
007222  013737  001510  004136  TEST3   MOV     @#FC,@#DPB.B+12 ;FC
007230  012737  007230  001110                  MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007236  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
007242  004037  025144                          JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
007246  013701  001514                          MOV     IC,R1           ;CYLINDER 1
007252  012737  007252  001110                  MOV     # ,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007260  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
007264  010137  004136          1$      MOV     R1,@#DPB B+12   ;DESIRED CYLINDER
007270  004037  025144                          JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
007274  006301                                  ASL     R1              ;MOVE TO NEXT CYLINDER
007276  020137  001512                          CMP     R1,@#LC         ;DONE?
007302  003770                                  BLE     1$              ;NO--LOOP
007304  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

                                        ;;*****************************************************************
007306                                  TST4
007306  000240                                  NOP
007310  033737  001434  001234                  BIT     @#BITS+<4*2>,TSTNMS ;DO THIS TEST?
007316  001002                                  BNE     64$             ;YES--BRANCH
007320  000137  007674                          JMP     TST5            ;NO--GO TO THE NEXT TEST
007324  012737  000004  001102  64$     MOV     #4,@#$TSTNM     ;SET UP TEST NUMBER AND
```

```
3606                                                    ;CLEAR THE ERROR FLAG ($ERFLG)
3607  007332  004737  024610                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
3608  007336  012737  007432  001110          MOV     #TEST4,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3609  007344  013777  001102  171570          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3610  007352  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
3611  007360  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
3612  007366  012737  007374  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
3613  007374  113737  001524  004134  1$.     MOVB    @#FS,@#DPB.B+10 ;FS
3614  007402  113737  001516  004135          MOVB    @#FT,@#DPB.B+11 ;FT
3615  007410  012737  007672  001252          MOV     #EXIT4,@#BYPASS ;GO TO EXIT4 ON ERROR
3616  007416  005002                          CLR     R2              ;CLEAR STALL SWITCH (NO STALL)
3617  007420  032737  010000  001220          BIT     #SW12,@#C.SWR   ;STALL REQUIRED?
3618  007426  001401                          BEQ     TEST4           ;NO--BRANCH
3619  007430  005102                          COM     R2              ;YES--SET SWITCH
3620  007432  013701  001510          TEST4   MOV     @#FC,R1         ;SET NC TO FC
3621  007436  005037  001336                  CLR     @#STALLO        ;START AT ZERO IF STALLS REQUIRED
3622  007442  012737  007442  001110          MOV     #.,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
3623  007450  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
3624  007454  010137  004136          1$      MOV     R1,@#DPB.B+12   ,NC
3625  007460  004037  025144                  JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
3626  007464  005702                          TST     R2              ;STALL?
3627  007466  001403                          BEQ     2$              ;NO--BRANCH
3628  007470  004037  026364                  JSR     R0,@#STALL      ;YES-GO TO STALL ROUTINE
3629  007474  001336                           WORD   STALLO          ;TIME POINTER
3630  007476  013737  001510  004136  2$      MOV     FC,@#DPB.B+12   ;FC
3631  007504  004037  025144                  JSR     R0,@#CALL B     ,GO EXECUTE THE COMMAND
3632  007510  005702                          TST     R2              ,STALL?
3633  007512  001413                          BEQ     3$              ,NO--BRANCH
3634  007514  004037  026364                  JSR     R0,@#STALL      ,YES--GO TO STALL ROUTINE
3635  007520  001336                           WORD   STALLO          ,TIME POINTER
3636  007522  005237  001336                  INC     @#STALLO        ,UPDATE THE TIME
3637  007526  023737  001362  001336          CMP     @#MXSTAL,@#STALLO ;TIME TO BIG?
3638  007534  003347                          BGT     1$              ;NO--BRANCH
3639  007536  005037  001336                  CLR     @#STALLO        ,YES--START OVER AT ZERO
3640  007542  063701  001514          3$      ADD     @#IC,R1         ,MOVE TO NEXT CYLINDER
3641  007546  020137  001512                  CMP     R1,@#LC         ,LAST CYLINDER COMPLETED?
3642  007552  003740                          BLE     1$              ;NO--BRANCH
3643  007554  013701  001512                  MOV     @#LC,R1         ,SET NC TO LC
3644  007560  012737  007560  001110          MOV     # ,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
3645  007566  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
3646  007572  010137  004136          4$      MOV     R1,@#DPB B+12   ,NC
3647  007576  004037  025144                  JSR     R0,@#CALL B     ,GO EXECUTE THE COMMAND
3648  007602  005702                          TST     R2              ,STALL?
3649  007604  001403                          BEQ     5$              ,NO--BRANCH
3650  007606  004037  026364                  JSR     R0,@#STALL      ,YES--GO TO STALL ROUTINE
3651  007612  001336                           WORD   STALLO          ,TIME POINTER
3652  007614  013737  001512  004136  5$      MOV     @#LC,@#DPB B+12 ,LC
3653  007622  004037  025144                  JSR     R0,@#CALL B     ,GO EXECUTE THE COMMAND
3654  007626  005702                          TST     R2              ,STALL?
3655  007630  001413                          BEQ     6$              ,NO--BRANCH
3656  007632  004037  026364                  JSR     R0,@#STALL      ,YES--GO TO STALL ROUTINE
3657  007636  001336                           WORD   STALLO          ,TIME POINTER
3658  007640  005237  001336                  INC     @#STALLO        ,UPDATE STALL TIME
3659  007644  023737  001362  001336          CMP     @#MXSTAL,@#STALLO ,TIME TOO BIG?
3660  007652  003347                          BGT     4$              ,NO--BRANCH
3661  007654  005037  001336                  CLR     @#STALLO        ,YES--SET STALL TIME BACK TO ZERO
```

```
3662  007660  163701  001514          6$      SUB     @#IC,R1         ;NEXT CYLINDER
3663  007664  020137  001510                  CMP     R1,@#FC         ;DONE?
3664  007670  002340                          BGE     4$              ;NO--BRANCH
3665  007672  000004                  EXIT4:  SCOPE                   ;LOOP
3666
3667                                  ;;**********************************************************************
3668                                  ;*TEST 5         CONVERGING/DIVERGING SEEK TEST
3669
3670                                  ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
3671                                  ;*      SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
3672                                  ;*      BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
3673                                  ;*      GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
3674                                  ;*      LESS THAN THE INITIAL VALUE OF "NC1".  AT THE COMPLETION OF
3675                                  ;*      EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
3676                                  ;*      ENSURE PROPER OPERATION.  "NC1" AND "NC2" DEFAULT TO
3677                                  ;*      "FC" AND "LC" RESPECTIVELY
3678
3679                                  ;;**********************************************************************
3680  007674                          TST5
3681  007674  000240                          NOP
3682  007676  033737  001436  001234          BIT     @#BITS+<5*2>,TSTNMS ;DO THIS TEST?
3683  007704  001002                          BNE     64$             ;YES--BRANCH
3684  007706  000137  010074                  JMP     TST6            ;NO--GO TO THE NEXT TEST
3685  007712  012737  000005  001102  64$     MOV     #5,@#$TSTNM     ;SET UP TEST NUMBER AND
3686                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
3687  007720  004737  024610                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
3688  007724  012737  010004  001110          MOV     #TEST5,@#$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
3689  007732  013777  001102  171202          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3690  007740  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
3691  007746  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
3692  007754  012737  007762  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
3693  007762  113737  001524  004134  1$      MOVB    @#FS,@#DPB.B+10 ;FS
3694  007770  113737  001516  004135          MOVB    @#FT,@#DPB B+11 ;FT
3695  007776  012737  010072  001252          MOV     #EXIT5,@#BYPASS ;GO TO EXIT5 ON ERROR
3696  010004  013701  001510          TEST5   MOV     @#FC,R1         ;START NC1 AT FC
3697  010010  013702  001512                  MOV     @#LC,R2         ;START NC2 AT LC
3698  010014  012737  010014  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3699  010022  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3700  010026  010137  004136          1$      MOV     R1,@#DPB B+12   ;NC1
3701  010032  004037  025144                  JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
3702  010036  010237  004136                  MOV     R2,@#DPB B+12   ;NC2
3703  010042  004037  025144                  JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
3704  010046  063701  001514                  ADD     @#IC,R1         ;NEXT NC1
3705  010052  163702  001514                  SUB     @#IC,R2         ;NEXT NC2
3706  010056  020137  001512                  CMP     R1,@#LC         ;DONE?
3707  010062  003003                          BGT     EXIT5           ;YES--BRANCH
3708  010064  020237  001510                  CMP     R2,@#FC ;?
3709  010070  002356                          BGE     1$              ;NO--BRANCH
3710  010072  000004                  EXIT5:  SCOPE                   ;LOOP
3711
3712                                  ;;**********************************************************************
3713                                  ;*TEST 6         SERVO ADDRESSING LOGIC NOISE GENERATOR
3714
3715                                  ;*      IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
3716                                  ;*      NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5   NOW "NC" IS UPDATED
3717                                  ;*      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.
                                        ;;*****************************************************************
010074                                  TST6:
010074  000240                          NOP
010076  033737  001440  001234          BIT     @#BITS+(6*2),TSTNMS ;DO THIS TEST?
010104  001002                          BNE     64$             ;YES--BRANCH
010106  000137  010340                  JMP     TST7            ;NO--GO TO THE NEXT TEST
010112  012737  000006  001102  64$:    MOV     #6,@#$TSTNM     ;SET UP TEST NUMBER AND
                                                                ;CLEAR THE ERROR FLAG ($ERFLG)
010120  004737  024610                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
010124  012737  010204  001110          MOV     #TEST6,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
010132  013777  001102  171002          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
010140  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
010146  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
010154  012737  010162  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
010162  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10 ;FS
010170  113737  001516  004135          MOVB    @#FT,@#DPB B+11 ;FT
010176  012737  010336  001252          MOV     #EXIT6,@#BYPASS ;GO TO EXIT6 ON ERROR
010204  013701  001510          TEST6   MOV     @#FC,R1         ;PICKUP "FC"
010210  013702  001512                  MOV     @#LC,R2         ;FORM LAST CYLINDER THAT
010214  162702  000005                  SUB     #5,R2           ;IS AVAILABLE FOR TESTING
010220  012737  010220  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
010226  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
010232  020102                  1$      CMP     R1,R2           ;LAST CYLINDER
010234  003040                          BGT     EXIT6           ;YES--BRANCH
010236  010137  004136                  MOV     R1,@#DPB.B+12   ;NC
010242  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
010246  062737  000004  004136          ADD     #4,@#DPB B+12   ;NC+4
010254  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
010260  162737  000003  004136          SUB     #3,@#DPB.B+12   ;NC+1
010266  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
010272  062737  000002  004136          ADD     #2,@#DPB.B+12   ;NC+3
010300  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
010304  162737  000001  004136          SUB     #1,@#DPB B+12   ;NC+2
010312  004037  025144                  JSR     R0,@#CALL B     ;GO EXECUTE THE COMMAND
010316  062737  000003  004136          ADD     #3,@#DPB.B+12   ;NC+5
010324  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
010330  063701  001514                  ADD     @#IC,R1
010334  000736                          BR      1$
010336  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'

                                        ;;*****************************************************************
010340                                  TST7:
```

```
3774  010340  000240                          NOP
3775  010342  033737  001442  001234          BIT     @#BITS+<7*2>,TSTNMS ;DO THIS TEST?
3776  010350  001002                          BNE     64$                 ;YES--BRANCH
3777  010352  000137  010720                  JMP     TST10               ,NO--GO TO THE NEXT TEST
3778  010356  012737  000007  001102  64$     MOV     #7,@#$TSTNM         ;SET UP TEST NUMBER AND
3779                                                                      ;CLEAR THE ERROR FLAG ($ERFLG)
3780  010364  004737  024610                  JSR     PC,LODPRM           ;LOAD THE PARMETERS FOR THE TEST
3781  010370  012737  010462  001110          MOV     #TEST7,@#$LPERR     ;SETUP THE LOOP ON ERROR ADDRESS
3782  010376  013777  001102  170536          MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3783  010404  013737  001506  001204          MOV     @#RPT,$TIMES        ;GET THE ITERATION COUNT
3784  010412  112737  000031  001115          MOVB    #25.,$ERMAX         ;MAX ERRORS ALLOWED FOR TEST
3785  010420  113737  001516  004135          MOVB    FT,DPB.B+11         ;LOAD STARTING TRACK ADDRESS
3786  010426  112737  000105  004106          MOVB    #SEEK,@#DPB.A+2     ;SEEK=COMMAND
3787  010434  112737  000173  004126          MOVB    #READHD,DPB.B+2     ;READ HEADER & DATA COMMAND
3788  010442  013704  034514                  MOV     RPADR,R4            ;UNIBUS ADDRESS OF THE RH11
3789  010446  012737  010716  001252          MOV     #EXIT7,BYPASS       ;ERROR TERMINATION ADDRESS
3790  010454  012737  010462  001106          MOV     #TEST7,$LPADR       ;SETUP THE LOOP ON TEST ADDRESS
3791  010462  012706  001100          TEST7   MOV     #STACK,SP           ,SETUP THE STACK POINTER
3792  010466  013737  001510  004136          MOV     FC,DPB B+12         ,INITIAL CYLINDER ADDRESS
3793  010474  023737  001510  001512          CMP     FC,LC               ,CYLINDER LIMITS THE SAME ?
3794  010502  001422                          BEQ     1$                  ,BR IF THEY ARE
3795  010504  004737  023510                  JSR     PC,$RAND            ;CYCYLE THE RANDOM NUMBER GENERATOR
3796  010510  013746  023606                  MOV     $HINUM,-(SP)        ,USE THE HIGH RANDOM NUMBER
3797  010514  005046                          CLR     -(SP)               ;UPPER DIVIDEND
3798  010516  013746  001512                  MOV     LC,-(SP)            ,FORM THE DIVISOR
3799  010522  005216                          INC     (SP)                ,INCREMENT
3800  010524  163716  001510                  SUB     FC,(SP)             ,SUBTRACT THE LOWER LIMIT
3801  010530  004737  023612                  JSR     PC,$DIV             ,DIVIDE
3802  010534  062637  004136                  ADD     (SP)+,DPB B+12      ,ADD THE REMAINDER TO THE INITIAL CYLINDER
3803  010540  005726                          TST     (SP)+               ,DISCARD THE QUOTENT
3804  010542  013737  004136  004116          MOV     DPB B+12,DPB.A+12   ;COPY NEW CYLINDER ADDRESS
3805  010550                          1$
3806  010550  012737  010550  001110          MOV     # ,$LPERR           ,SETUP THE ERROR LOOP ADDRESS
3807  010556  012706  001100                  MOV     #STACK,SP           ,LOAD THE STACK POINTER
3808  010562  004037  025032                  JSR     R0,@#CALL A         ,GO EXECUTE THE COMMAND
3809  010566  012737  010566  001110          MOV     # ,$LPERR           ,SETUP THE ERROR LOOP ADDRESS
3810  010574  012706  001100                  MOV     #STACK,SP           ;LOAD THE STACK POINTER
3811  010600  113764  004104  000010          MOVB    DPB A,RPCS2(R4)     ,SELECT THE DRIVE
3812  010606  016446  000020                  MOV     RPLA(R4),-(SP)      ,GET THE LOOK AHEAD REGISTER
3813  010612  006316                          ASL     (SP)                ;ALIGN THE SECTOR ADDRESS
3814  010614  006316                          ASL     (SP)                ,ALIGN THE SECTOR ADDRESS
3815  010616  000316                          SWAB    (SP)                ,PUT ADDRESS IN LOWER BYTE
3816  010620  105766  000001                  TSTB    1(SP)               ,IN THE 1ST 20% OF SECTOR ?
3817  010624  001401                          BEQ     2$                  ,BR IF YES
3818  010626  105216                          INCB    (SP)                ,INCREMENT THE SECTOR ADDRESS
3819  010630  105216                  2$      INCB    (SP)                ;INCREMENT THE SECTOR ADDRESS
3820  010632  112637  004174                  MOVB    (SP)+,DTADPB+10     ,LOAD THE DPB
3821  010636  013746  001630                  MOV     PRMLMT+22,-(SP)     ,PUT LAST SECTOR ADDRESS ON THE STACK
3822  010642  005216                          INC     (SP)                ,INCREMENT IT
3823  010644  122637  004174                  CMPB    (SP)+,DTADPB+10     ,NEW SECTOR ADDRESS TOO LARGE ?
3824  010650  103007                          BHIS    4$                  ;BR IF NOT
3825  010652  103403                          BLO     3$                  ,BR IF ADDRESS IS 2 GREATER
3826  010654  105037  004174                  CLRB    DTADPB+10           ,RESET TO SECTOR ADDRESS 0
3827  010660  000403                          BR      4$                  ,CONTINUE
3828  010662  112737  000001  004174  3$      MOVB    #1,DTADPB+10        ,RESET ADDRESS TO SECTOR 1
3829  010670                          4$
```

```
3830  010670  004037  025144                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
3831  010674  105237  004135                  INCB    DPB.B+11        ;INCREMENT THE TRACK ADDRESS
3832  010700  123737  004135  001520          CMPB    DPB.B+11,LT     ;MAXIMUM ?
3833  010706  101403                          BLOS    EXIT7           ;BR IF NOT
3834  010710  113737  001516  004135          MOVB    FT,DPB.B+11     ;RELOAD STARTING TRACK ADDRESS
3835  010716  000004                  EXIT7   SCOPE                   ;LOOP ?
3836
3837
3838                                  ;;*****************************************************************
3839                                  ;*TEST 10        SERVO SETTLE DOWN TEST
3840
3841                                  ;*      THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT
3842                                  ;*      THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE.
3843                                  ;*      RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+1C'
3844                                  ;*      ('NC1' STARTS AT VALUE 'FC').  AT THE COMPLETION OF 1000 (10) SEEKS,
3845                                  ;*      'NC1' IS INCREMENTED BY VALUE '1C' AND THE SEQUENCE IS REPEATED
3846                                  ;*      THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'.
3847                                  ;*
3848                                  ;*      WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD
3849                                  ;*      REGISTER (RPLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING  INTO
3850                                  ;*      POSITION   THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
3851                                  ;*      FOR THAT SECTOR.  IF THE DRIVE'S POSITIONER HAS NOT SETTLED DOWN OR
3852                                  ;*      IF THE POSITIONER IS NOT ON CYLINDER (IF THE DRIVE IS AN RP04, THE
3853                                  ;*      OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 US), THE DRIVE
3854                                  ;*      WILL REPORT A 'WRU' ERROR.  (RP05/6'S MAY ALSO REPORT 'NHS' ERROR UNDER
3855                                  ;*      ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED
3856                                  ;*      CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PACK WITH
3857                                  ;*      MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE.
3858                                  ;*
3859                                  ;*      THIS TEST USES THE EXTENTION BITS IN THE LOOK-AHEAD REGISTER TO DETERMINE
3860                                  ;*      WHETHER OR NOT IT CAN PICK UP THE SECTOR ROTATING INTO POSITION.  THE
3861                                  ;*      TEST IS OPTIMIZED SUCH THAT IF THE DRIVE SIGNALS SEEK DONE WITHIN
3862                                  ;*      THE FIRST 80% OF THE SECTOR CURRENTLY UNDER THE HEAD, THE TEST WILL
3863                                  ;*      TRY TO ADDRESS THE NEXT SECTOR  BASED ON OBSERVATION, THE PROGRAM
3864                                  ;*      IS ABLE TO START THE OPERATION WITHOUT LOSING A REVOLUTION MOST OF
3865                                  ;*      THE TIME
3866                                  ;*
3867                                  ;*      THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
3868                                  ;*      HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS  THE NECESSARY TIME
3869                                  ;*      TIME DEPENDENT PARAMETERS OCCUR FREQUENTLY ENOUGH WITHIN THE REQUIRED
3870                                  ;*      RANGE TO PERMIT THIS TEST TO BE EFFECTIVE
3871
3872                                  ;;*****************************************************************
3873  010720                          TST10
3874  010720  000240                          NOP
3875  010722  033737  001444  001234          BIT     @#BITS+<10*2>,TSTNMS ;DO THIS TEST?
3876  010730  001002                          BNE     64$             ;YES--BRANCH
3877  010732  000137  012024                  JMP     TST11           ;NO--GO TO THE NEXT TEST
3878  010736  012737  000010  001102  64$     MOV     #10,@#$TSTNM    ;SET UP TEST NUMBER AND
3879                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
3880  010744  004737  024610                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
3881  010750  012737  011126  001110          MOV     #TEST10,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3882  010756  013777  001102  170156          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3883  010764  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
3884  010772  112737  000031  001115          MOVB    #25 ,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
3885  011000  012737  011006  001106          MOV     #1$,$LPADR      ;SETUP THE LOOP ADDRESS
```

```
3886  011006                          1$
3887  011006  112737  000105  004106          MOVB    #SEEK,@#DPB.A+2 ;SEEK=COMMAND
3888  011014  112737  000161  004166          MOVB    #WRITE,DTADPB+2 ;COMMAND
3889  011022  113737  001516  004175          MOVB    FT,DTADPB+11    ;TRACK ADDRESS FOR THE WRITE
3890  011030  013737  001510  004116          MOV     FC,DPB.A+12     ;CYLINDER ADDRESS FOR THE SEEK
3891  011036  013737  001510  004176          MOV     FC,DTADPB+12    ;CYLINDER ADDRESS FOR THE WRITE
3892  011044  013737  001510  001532          MOV     FC,NC1          ;STARTING CYLINDER
3893  011052  013737  001514  001350          MOV     IC,DELTA        ;CYLINDER INCREMENT VALUE
3894  011060  012737  176000  004170          MOV     #-(256.*4.),DTADPB+4  ;WORD COUNT
3895  011066  012737  047712  004172          MOV     #BUFFER,DTADPB+6  ;BUFFER ADDRESS
3896  011074  005000                          CLR     R0              ;PATTERN POINTER (WC PATTERN)
3897  011076  004737  030360                  JSR     PC,SETBUF       ;LOAD THE WRITE BUFFER
3898  011102  005001                          CLR     R1              ;CLEAR REGISTER
3899  011104  113701  004104                  MOVB    DPB.A,R1        ;LOAD DRIVE ADDRESS
3900  011110  013704  034514                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF THE RH11
3901  011114  004737  042516                  JSR     PC,CLRQUE       ;CLEAR THE OPERATION QUEUES
3902  011120  012737  012022  001252          MOV     #EXIT10,BYPASS  ;ERROR EXIT FROM TEST
3903  011126                          TEST10
3904  011126  012737  011126  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3905  011134  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3906  011140  012737  000340  177776          MOV     #PR7,@#PS       ;SET PRIORITY TO 7
3907  011146  005737  001244                  TST     CLKSTA          ;SEE WHICH CLOCK ON SYSTEM
3908  011152  001415                          BEQ     3$              ;BR IF NO CLOCK
3909  011154  100405                          BMI     1$              ;BR IF KW11-L CLOCK
3910  011156  017746  170212                  MOV     @PKV,-(SP)      ;SAVE THE VECTOR
3911  011162  013746  001374                  MOV     PKV,-(SP)       ;SAVE THE VECTOR ADDRESS
3912  011166  000404                          BR      2$              ;CONTINUE
3913  011170  017746  170212          1$      MOV     @LKV,-(SP)      ;SAVE THE 'L' CLOCK VECTOR
3914  011174  013746  001406                  MOV     LKV,-(SP)       ;SAVE THE VECTOR ADDRESS
3915  011200  012776  011756  000000  2$      MOV     #TST10B,@(SP)   ;CHANGE THE VECTOR
3916  011206  012777  027052  023302  3$      MOV     #DORTI,@RPVEC   ;CHANGE THE RP04/RP05 VECTOR
3917  011214  012737  000010  001344          MOV     #8.,.SEKTMR     ;LOAD THE SEEK TIMER
3918  011222  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;INIT THE MASSBUS
3919  011230  110164  000010                  MOVB    R1,RPCS2(R4)    ;RESELECT THE DRIVE
3920  011234  013764  004116  000034          MOV     DPB.A+12,RPCA(R4)  ;LOAD THE CYLINDER ADDRESS
3921  011242  013737  004116  001270          MOV     DPB.A+12,CYL.DS ;CYLINDER ADDRESS FOR ERROR MESSAGE
3922  011250  112764  000105  000000          MOVB    #SEEK,RPCS1(R4) ;START THE SEEK
3923  011256  005037  177776                  CLR     @#PS            ;CLEAR THE PRIORITY
3924  011262  105764  000012          4$      TSTB    RPDS1(R4)       ;HAS THE DRIVE FINISHED ?
3925  011266  100402                          BMI     5$              ;BR IF IT HAS
3926  011270  000001                          WAIT                    ;WAIT FOR THE OPERATION TO COMPLETE
3927  011272  000773                          BR      4$              ;CONTINUE
3928  011274  012737  000340  177776  5$      MOV     #PR7,@#PS       ;CHANGE PRIORITY TO MAX
3929  011302  032764  040000  000012          BIT     #BIT14,RPDS1(R4)  ;ERROR ?
3930  011310  001412                          BEQ     6$              ;BR IF NOT
3931  011312  012702  004104                  MOV     #DPB.A,R2       ;DPB POINTER
3932  011316  004737  042034                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
3933  011322  104023                          ERROR   23              ;ERROR DURING SEEK
3934  011324  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;INIT THE MASSBUS
3935  011332  110164  000010                  MOVB    R1,RPCS2(R4)    ;RESELECT THE DRIVE
3936  011336  012777  037376  023152  6$      MOV     #ISR,@RPVEC     ;SETUP THE RP04/RP05 VECTOR
3937  011344  005737  001244                  TST     CLKSTA          ;WHICH CLOCK
3938  011350  001405                          BEQ     TST10A          ;BR IF NONE
3939  011352  016676  000002  000000          MOV     2(SP),@(SP)     ;RELOAD THE CLOCK VECTOR
3940  011360  062706  000004                  ADD     #4,SP           ;CORRECT THE STACK POINTER
3941  011364                          TST10A
```

```
3942  011364  012737  011364  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3943  011372  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3944  011376  110164  000010                  MOVB    R1,RPCS2(R4)    ;SELECT THE DRIVE
3945  011402  016446  000020                  MOV     RPLA(R4),-(SP)  ;GET THE LOOK AHEAD REGISTER
3946  011406  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
3947  011410  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
3948  011412  000316                          SWAB    (SP)            ;PUT ADDRESS IN LOWER BYTE
3949  011414  122766  000300  000001          CMPB    #300,1(SP)      ;IN THE LAST 20% OR SECTOR ?
3950  011422  001001                          BNE     2$              ;BR IF NOT
3951  011424  105216                          INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
3952  011426  105216                  2$      INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
3953  011430  112637  004174                  MOVB    (SP)+,DTADPB+10 ;LOAD THE DPB
3954  011434  013746  001630                  MOV     PRMLMT+22,-(SP)  ;PUT MAXIMUM SECTOR ADDRESS ON THE STACK
3955  011440  005216                          INC     (SP)            ,INCREMENT PAST THE MAXIMUM ADDRESS
3956  011442  122637  004174                  CMPB    (SP)+,DTADPB+10 ;NEW SECTOR ADDRESS TOO LARGE ?
3957  011446  101007                          BHI     4$              ;BR IF NOT
3958  011450  103403                          BLO     3$              ,BR IF ADDRESS IS 2 GREATER THAN MAXIMUM
3959  011452  105037  004174                  CLRB    DTADPB+10       ,RESET TO SECTOR ADDRESS 0
3960  011456  000403                          BR      4$              ;CONTINUE
3961  011460  112737  000001  004174  3$      MOVB    #1,DTADPB+10    ;RESET ADDRESS TO SECTOR 1
3962  011466  012703  004170          4$      MOV     #DTADPB+4,R3    ;POINTER
3963  011472  012764  000111  000000          MOV     #DRVCLR,RPCS1(R4)  ,CLEAR THE DRIVE
3964  011500  012364  000002                  MOV     (R3)+,RPWC(R4)  ;LOAD THE WORD COUNT
3965  011504  012364  000004                  MOV     (R3)+,RPBA(R4)  ;LOAD THE BUFFER ADDRESS
3966  011510  012364  000006                  MOV     (R3)+,RPDA(R4)  ;LOAD THE TRACK/SECTOR ADDR
3967  011514  005037  004202                  CLR     DTADPB+16       ;RESET 'DONE' INDICATOR
3968  011520  012737  004164  034426          MOV     #DTADPB,TRNSWT  ;LOAD 'TRANSFER' DPB ADDRESS
3969  011526  010137  034500                  MOV     R1,DTUW         ,ADDRESS OF DRIVE TRANSFERING
3970  011532  112761  000001  034356          MOVB    #1,DRVACT(R1)   ,SET DRIVE ACTIVE INDICATOR
3971  011540  006301                          ASL     R1              ,SHIFT DRIVE ADDRESS
3972  011542  012761  001750  034460          MOV     #1000.,TIMER(R1)  ,SETUP THE OPERATION TIMER
3973  011550  006201                          ASR     R1              ;RESTORE R1
3974  011552  013764  004166  000000          MOV     DTADPB+2,RPCS1(R4)  ;START THE OPERATION
3975  011560  005037  177776                  CLR     @#PS            ,CLEAR THE PRIORITY
3976  011564  004037  025544                  JSR     R0,DRVCL1       ;WAIT FOR OPERATION TO COMPLETE
3977  011570  023727  001346  001750  5$      CMP     SEKCNT,#1000    ,FINISHED SEEKS ?
3978  011576  001026                          BNE     6$              ,BR IF NOT
3979  011600  005037  001346                  CLR     SEKCNT          ;CLEAR THE SEEK COUNT
3980  011604  063737  001514  001532          ADD     IC,NC1          ,ADD THE INCREMENT
3981  011612  023737  001532  001512          CMP     NC1,LC          ;EXCEEDED THE CYLINDER LIMIT ?
3982  011620  103100                          BHIS    EXIT10          ;BR IF IT HAS
3983  011622  013737  001512  001350          MOV     LC,DELTA        ,GET THE NEXT 'ZONE' ADDRESS
3984  011630  163737  001532  001350          SUB     NC1,DELTA       ,CHECK THE DIFFERENCE
3985  011636  023737  001514  001350          CMP     IC,DELTA        ,DIFFERENCE GREATER THAN THE INCREMENT ?
3986  011644  101003                          BHI     6$              ,BR IF IT IS
3987  011646  013737  001514  001350          MOV     IC,DELTA        ;USE THE ICREMENT PARAMETER
3988  011654  005237  001346          6$      INC     SEKCNT          ,COUNT THE NEXT SEEK
3989  011660  023737  001510  001512          CMP     FC,LC           ,BEGINNING AND ENDING CYLINDERS THE SAME ?
3990  011666  001002                          BNE     7$              ,BR IF NOT
3991  011670  000137  011126                  JMP     TEST10          ,BR IF THEY ARE
3992  011674  013737  001532  004116  7$      MOV     NC1,DPB A+12    ;RESET THE CYLINDER ADDRESS
3993  011702  004737  023510                  JSR     PC,$RAND        ,CYCLE THE RANDOM NUMBER GENERATOR
3994  011706  013746  023606                  MOV     $HINUM,-(SP)    ,USE THE HIGH RANDOM NUMBER
3995  011712  005046                          CLR     -(SP)           ,CLEAR THE UPPER DIVIDEND
3996  011714  013746  001350                  MOV     DELTA,-(SP)     ,FORM THE DIVISOR
3997  011720  005216                          INC     (SP)            ,INCREMENT
```

```
 3998  011722  004737  023612                  JSR     PC,$DIV          ;DIVIDE
 3999  011726  062637  004116                  ADD     (SP)+,DPB A+12   ;ADD THE REMAINDER TO THE INITIAL CYLINDER
 4000  011732  005726                          TST     (SP)+            ;DISCARD THE QUOTENT
 4001  011734  023737  004116  004176          CMP     DPB.A+12,DTADPB+12  ;SAME CYLINDER SELECTED AS LAST TIME ?
 4002  011742  001754                          BEQ     7$               ;BR IF IT WAS
 4003  011744  013737  004116  004176          MOV     DPB.A+12,DTADPB+12  ;COPY NEW CYLINDER ADDRESS
 4004  011752  000137  011126                  JMP     TEST10           ;CONTINUE
 4005  011756  005337  001344          TST10B  DEC     SEKTMR           ;DECREMENT THE SEEK TIMER
 4006  011762  001016                          BNE     1$               ;CONTINUE IF NOT DONE
 4007  011764  012702  004104                  MOV     #DPB A,R2        ;DPB ADDRESS
 4008  011770  004737  042034                  JSR     PC,SVRH11        ;SAVE THE REGISTERS
 4009  011774  104024                          ERROR   24               ;TIMEOUT DURING SEEK
 4010  011776  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;INIT THE MASSBUS
 4011  012004  110164  000010                  MOVB    R1,RPCS2(R4)     ;RESELECT THE DRIVE
 4012  012010  016676  000002  000000          MOV     2(SP),@(SP)      ;RESTORE THE CLOCK VECTOR ADDRESS
 4013  012016  000401                          BR      EXIT10           ;ABORT THE TEST
 4014  012020  000002                  1$      RTI                      ;RETURN
 4015  012022  000004                  EXIT10: SCOPE                    ;LOOP ?
 4016
 4017                                  ;;*****************************************************************
 4018                                  ;*TEST 11        ALL SEEKS TEST
 4019
 4020                                  ;*      THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER
 4021                                  ;*      TO ALL OTHER CYLINDERS
 4022                                  ;*
 4023                                  ;*      BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER
 4024                                  ;*      BETWEEN 'FC' AND 'LC' FROM CYINDER 'FC'   THE BEGINNING CYLINDER
 4025                                  ;*      ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER
 4026                                  ;*      ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC'.   THE SEQUENCE
 4027                                  ;*      CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED
 4028
 4029                                  ;;*****************************************************************
 4030  012024                  TST11
 4031  012024  000240                          NOP
 4032  012026  033737  001446  001234          BIT     @#BITS+(11*2),TSTNMS ;DO THIS TEST?
 4033  012034  001002                          BNE     64$              ;YES--BRANCH
 4034  012036  000137  012244                  JMP     TST12            ;NO--GO TO THE NEXT TEST
 4035  012042  012737  000011  001102  64$     MOV     #11,@#$TSTNM     ;SET UP TEST NUMBER AND
 4036                                                                   ;CLEAR THE ERROR FLAG ($ERFLG)
 4037  012050  004737  024610                  JSR     PC,LODPRM        ;LOAD THE PARMETERS FOR THE TEST
 4038  012054  012737  012164  001110          MOV     #TEST11,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
 4039  012062  013777  001102  167052          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 4040  012070  013737  001506  001204          MOV     @#RPT,$TIMES     ;GET THE ITERATION COUNT
 4041  012076  112737  000031  001115          MOVB    #25.,$ERMAX      ;MAX ERRORS ALLOWED FOR TEST
 4042  012104  012737  012112  001106          MOV     #1$,$LPADR       ;SETUP THE LOOP ADDRESS
 4043  012112  113737  001524  004134  1$:     MOVB    FS,DPB.B+10      ;SECTOR ADDRESS
 4044  012120  113737  001524  004154          MOVB    FS,DPB.C+10      ;SECTOR ADDRESS
 4045  012126  113737  001516  004135          MOVB    FT,DPB B+11      ;TRACK ADDRESS
 4046  012134  113737  001516  004155          MOVB    FT,DPB.C+11      ;TRACK ADDRESS
 4047  012142  013737  001510  004136          MOV     FC,DPB.B+12      ;STARTING CYLINDER ADDRESS
 4048  012150  013737  001510  004156          MOV     FC,DPB.C+12      ;STARTING CYLINDER ADDRESS
 4049  012156  012737  012242  001252          MOV     #EXIT11,BYPASS   ;TEST ABORT EXIT
 4050  012164  012706  001100          TEST11: MOV     #STACK,SP        ;SETUP THE STACK POINTER
 4051  012170                          1$:
 4052  012170  004037  025334                  JSR     R0,@#CALL C      ;GO EXECUTE THE COMMAND
 4053  012174  004037  025144                  JSR     R0,@#CALL B      ;GO EXECUTE THE COMMAND
```

```
 4054  012200  063737  001514  004156          ADD     IC,DPB C+12     ;INCREMENT THE ENDING CYLINDER ADDRESS
 4055  012206  023737  001512  004156          CMP     LC,DPB C+12     ;CHECK IF EXCEEDING MAXIMUM
 4056  012214  002365                          BGE     1$              ;BR IF NOT
 4057  012216  013737  001510  004156          MOV     FC,DPB C+12     ;RESET ENDING CYLINDEP ADDRESS
 4058  012224  063737  001514  004136          ADD     IC,DPB B+12     ;INCREMENT THE STARTING ADDRESS
 4059  012232  023737  001512  004136          CMP     LC,DPB B+12     ;EXCEEDING MAXIMUM ?
 4060  012240  002353                          BGE     1$              ;BR IF NOT
 4061  012242  000004                  EXIT11  SCOPE                   ;LOOP ?
 4062
```

```
4063
4064                                    SBTTL  *** TIMING TESTS ***
4065

        ,/*  /* ./* ./*  /*  /* ./* ./*  /*  /* :/* :/* :/* :/* :/* :/* ./* :/* .
        ,/*  /*  /*  /* ./*  /*  /*  /* ./* ./* ./* :/* :/* :/* :/* :/* :/* :/* :
        ,/* ./* ./*  /* ./* ./* :/*  /* ./* :/* :/* :/* :/* :/* :/* ./* :/* :/* :

        ,*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

        ,/  */  */  */  */  */  */  */: */  */  */: */: */: */: */: */. */. *
        ,/  */ ./*  */  */  */  */. */. */: */: */: */: */: */: */: */: */: *
        ,/  */  */  */  */  */  */  */  */  */. */  */. */. */: */: */: */: */. *

4083
4084                                    ,,**********************************************************************
4085                                    ;*TEST 12       ROTATIONAL SPEED TIMING TEST
4086
4087                                    ;*      THIS TEST WILL START A SEARCH TO CYLINDER 0, TRACK 0, SECTOR
4088                                    ,*      0   AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS SET AGAIN
4089                                    ,*      AND THE OPERATION IS TIMED.  THIS PROCEDURE IS REPEATED 10
4090                                    ,*      TIMES THEN THE AVERAGE TIME IS CALCULATED AND CHECKED TO
4091                                    ;*      ENSURE IT IS WITHIN TOLERANCE:
4092                                    ;*              16.67 MS/REV + OR - 2% IF 60HZ
4093                                    ,*              16.67 MS/REV + OR - 2.5% IF 50HZ
4094
4095                                    ;,**********************************************************************
4096  012244                            TST12:
4097  012244  000240                           NOP
4098  012246  033737  001450  001234           BIT     @#BITS+<12*2>,TSTNMS ;DO THIS TEST?
4099  012254  001002                           BNE     64$                  ;YES--BRANCH
4100  012256  000137  013002                   JMP     TST13                ;NO--GO TO THE NEXT TEST
4101  012262  012737  000012  001102  64$      MOV     #12,@#$TSTNM         ;SET UP TEST NUMBER AND
4102                                                                        ;CLEAR THE ERROR FLAG ($ERFLG)
4103  012270  004737  024610                   JSR     PC,LODPRM            ;LOAD THE PARMETERS FOR THE TEST
4104  012274  012737  012244  001110           MOV     #TST12,@#$LPERR      ;SETUP THE ERROR LOOP ADDRESS
4105  012302  013777  001102  166632           MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4106  012310  013737  001506  001204           MOV     @#RPT,$TIMES         ;GET THE ITERATION COUNT
4107  012316  112737  000031  001115           MOVB    #25.,$ERMAX          ;MAX ERRORS ALLOWED FOR TEST
4108  012324  005737  001244                   TST     @#CLKSTA             ;KW11-P CLOCK?
4109  012330  003002                           BGT     1$                   ;YES--START TEST
4110  012332  000137  013002                   JMP     TST13                ;NO--GO TO NEXT TEST
4111  012336  012737  012336  001106  1$       MOV     #1$,$LPADR           ;SETUP LOOP ADDRESS
4112  012344  004037  026670                   JSR     R0,@#SRCH00          ;DO A MASSBUS INIT & RECAL
4113  012350  000402                           BR      2$                   ;RETURN HERE IF NO ERROR
4114  012352  000137  013000                   JMP     EXIT12               ;RETURN HERE IF ERROR
4115  012356  013764  001510  000034  2$       MOV     @#FC,RPCA(R4)        ;FC
4116  012364  013746  001524                   MOV     @#FS,-(SP)           ,FS
4117  012370  113766  001516  000001           MOVB    @#FT,1(SP)           ;FT
4118  012376  012664  000006                   MOV     (SP)+,RPDA(R4)       ,LOAD FT/FS
```

```
4119  012402  012737  013000  001206          MOV     #EXIT12,$ESCAPE ;,ESCAPE TO EXIT12 ON ERROR
4120  012410  005005                          CLR     R5              ;COUNT UP
4121  012412  012703  002774                  MOV     #T7A,R3         ;60HZ PARAMETERS
4122  012416  032737  000100  001220          BIT     #SW06,@#C.SWR   ;60 HZ?
4123  012424  001402                          BEQ     TEST12          ;YES--BRANCH
4124  012426  012703  003004                  MOV     #T7B,R3         ;NO--50 HZ PARAMETERS
4125  012432  012706  001100          TEST12. MOV     #STACK,SP       ;SETUP STACK
4126  012436  012701  000012                  MOV     #10.,R1         ;TIME 10 SEARCHES
4127  012442  004737  027054                  JSR     PC,@#STRTMR     ;INITIALIZE THE TIMERS
4128  012446  012777  012654  166720          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
4129  012454  012777  027052  022034          MOV     #DORTI,@RPVEC   ;SETUP RP04/5/6 VECTOR
4130  012462  005077  166714          1$:     CLR     @PKB            ;START COUNTING AT ZERO
4131  012466  012777  000131  166704          MOV     #131,@PKCS      ;INT.EN., COUNT UP AT 100KHZ
4132  012474  012714  000131                  MOV     #SEARCH,(R4)    ;START A SEARCH
4133  012500  000001                          WAIT                    ;WAIT ON INTERRUPT
4134  012502  042777  000101  166670          BIC     #101,@PKCS      ;STOP THE CLOCK
4135  012510  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ERROR?
4136  012516  001415                          BEQ     2$              ;NO--BRANCH
4137  012520  104412                          SAVREG                  ;SAVE R0-R5
4138  012522  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4139  012526  004737  042034                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4140  012532  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4141  012540  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ,SELECT DRIVE
4142  012546  104413                          RESREG                  ;RESTORE R0-R5
4143  012550  104017                          ERROR   17
4144  012552  005077  166624          2$      CLR     @PKB            ;START THE COUNT AT ZERO
4145  012556  012714  000131                  MOV     #SEARCH,(R4)    ,START A SEARCH
4146  012562  012777  000131  166610          MOV     #131,@PKCS      ,START THE CLOCK
4147  012570  000001                          WAIT                    ;WAIT ON INTERRUPT
4148  012572  042777  000101  166600          BIC     #101,@PKCS      ,STOP THE CLOCK
4149  012600  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ,IS "ERR=1"?
4150  012606  001415                          BEQ     3$              ;NO--BRANCH
4151  012610  104412                          SAVREG                  ;SAVE R0-R5
4152  012612  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4153  012616  004737  042034                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4154  012622  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ,MASSBUS CLEAR
4155  012630  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4156  012636  104413                          RESREG                  ;RESTORE R0-R5
4157  012640  104017                          ERROR   17              ,DISK ERROR OCCURRED
4158  012642  004737  027120          3$      JSR     PC,@#COUNT      ;UPDATE THE COUNT
4159  012646  005301                          DEC     R1              ;DONE?
4160  012650  003304                          BGT     1$              ;NO--BRANCH
4161  012652  000424                          BR      8$              ;YES--GO TO THE EXIT
4162  012654  042777  000101  166516  7$      BIC     #101,@PKCS      ,STOP THE CLOCK
4163  012662  005037  177776                  CLR     @#PS            ,DROP THE PRIORITY
4164  012666  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
4165  012670  005726                          TST     (SP)+           ,POP THE PS FROM THE STACK
4166  012672  104412                          SAVREG                  ,SAVE R0-R5
4167  012674  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4168  012700  004737  042034                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4169  012704  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4170  012712  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4171  012720  104413                          RESREG                  ,RESTORE R0-R5
4172  012722  104020                          ERROR   20              ,CLOCK OVERFLOWED
4173  012724                          8$
4174  012724  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ,MASSBUS INIT
```

```
4175  012732  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4176  012740  004737  024074                  JSR     PC,@#ST.CLK        ;INITIALIZE THE CLOCK
4177  012744  012777  037376  021544          MOV     #ISR,@RPVEC        ;RESTORE RH11/RP04/5/6 INT  VECTOR
4178  012752  032737  000100  001220          BIT     #SW06,@#C.SWR      ;60 HZ?
4179  012760  001004                          BNE     9$                 ;NO -- BRANCH
4180  012762  004037  027252                  JSR     R0,@#TYPTIM        ;GO TYPE THE TIMES
4181  012766  002774                          T7A                        ;POINTER
4182  012770  000403                          BR      EXIT12             ;GO TO EXIT
4183  012772                          9$:
4184  012772  004037  027252                  JSR     R0,@#TYPTIM        ;GO TYPE THE TIMES
4185  012776  003004                          T7B                        ;POINTER
4186  013000  000004                  EXIT12: SCOPE                      ;LOOP ?
4187
4188
4189                                  ;;**********************************************************************
4190                                  ;*TEST 13        ONE CYLINDER SEEK TIMING TEST
4191
4192                                  ;*      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
4193                                  ;*      CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
4194                                  ;*      CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'   THE
4195                                  ;*      TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
4196                                  ;*      EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK
4197                                  ;*      THE TIME MUST BE LESS THAN 10MS.
4198
4199                                  ;;**********************************************************************
4200  013002                          TST13:
4201  013002  000240                          NOP
4202  013004  033737  001452  001234          BIT     @#BITS+<13*2>,TSTNMS ;DO THIS TEST?
4203  013012  001002                          BNE     64$                ;YES--BRANCH
4204  013014  000137  013446                  JMP     TST14              ;NO--GO TO THE NEXT TEST
4205  013020  012737  000013  001102  64$:    MOV     #13,@#$TSTNM       ;SET UP TEST NUMBER AND
4206                                                                     ;CLEAR THE ERROR FLAG ($ERFLG)
4207  013026  004737  024610                  JSR     PC,LODPRM          ;LOAD THE PARMETERS FOR THE TEST
4208  013032  012737  013002  001110          MOV     #TST13,@#$LPERR    ;SETUP THE ERROR LOOP ADDRESS
4209  013040  013777  001102  166074          MOV     $TSTNM,@DISPLAY    ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4210  013046  013737  001506  001204          MOV     @#RPT,$TIMES       ;GET THE ITERATION COUNT
4211  013054  112737  000031  001115          MOVB    #25.,$ERMAX        ;MAX ERRORS ALLOWED FOR TEST
4212  013062  005737  001244                  TST     @#CLKSTA           ;KW11-P CLOCK?
4213  013066  003002                          BGT     1$                 ;YES--START TEST
4214  013070  000137  013446                  JMP     TST14              ;NO--GO TO NEXT TEST
4215  013074  012737  013074  001106  1$:     MOV     #1$,$LPADR         ;SETUP THE LOOP ADDRESS
4216  013102  004037  026670                  JSR     R0,@#SRCH00        ;DO A MASSBUS INIT.  AND RECAL
4217  013106  000402                          BR      2$                 ;NO ERROR RETURN
4218  013110  000137  013444                  JMP     EXIT13             ;ERROR RETURN--SCOPE LOOP CALL
4219  013114  012703  003014          2$:     MOV     #T10,R3            ;PARAMETER POINTER
4220  013120  012737  013444  001206          MOV     #EXIT13,$ESCAPE    ;;ESCAPE TO EXIT13 ON ERROR
4221  013126  012706  001100          TEST13: MOV     #STACK,SP          ;SETUP STACK
4222  013132  013737  001510  004176          MOV     FC,@#DTADPB+12     ;START WITH BEGINNING CYLINDER
4223  013140  005237  004176                  INC     DTADPB+12          ;INCREMENT THE BEGINNING CYLINDER
4224  013144  005005                          CLR     R5                 ;SET THE UP/DOWN SWITCH TO UP
4225  013146  004737  027054                  JSR     PC,@#STRTMR        ;INITIALIZE THE TIMERS
4226  013152  012777  013340  166214          MOV     #7$,@PKV           ;SETUP INCASE OF OVERFLOW
4227  013160  012777  027052  021330          MOV     #DORTI,@RPVEC      ;SET RP04/5/6 VECTOR
4228  013166  005077  166210          1$:     CLR     @PKB               ;START THE COUNTER AT ZERO
4229  013172  013764  004176  000034          MOV     @#DTADPB+12,RPCA(R4) ;LOAD DESIRED CYLINDER
4230  013200  012714  000105                  MOV     #SEEK,(R4)         ;START A SEEK
```

```
4231  013204  012777  000131  166166          MOV     #131,@PKCS      ;START THE CLOCK
4232  013212  000001                          WAIT                    ;WAIT ON INTERRUPT
4233  013214  002777  000101  166156          BIC     #101,@PKCS      ;STOP THE CLOCK
4234  013222  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ANY DISK ERRORS?
4235  013230  001415                          BEQ     2$              ;NO--BRANCH
4236  013232  104412                          SAVREG                  ;SAVE R0-R5
4237  013234  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4238  013240  004737  042034                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4239  013244  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4240  013252  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4241  013260  104413                          RESREG                  ;RESTORE R0-R5
4242  013262  104017                          ERROR   17              ;REPORT THE ERROR
4243  013264  004737  027120          2$      JSR     PC,@#COUNT      ;COUNT THIS SEEKS TIME
4244  013270  004737  026446                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
4245  013274  005705                          TST     R5              ;UP OR DOWN?
4246  013276  001011                          BNE     4$              ;DOWN--BRANCH
4247  013300  005237  004176          3$      INC     @#DTADPB+12     ;MOVE TO NEXT CYLINDER
4248  013304  023737  004176  001512          CMP     @#DTADPB+12,LC  ;OUT OF CYLINDERS?
4249  013312  002725                          BLT     1$              ;NO--GO DO THE NEXT SEEK
4250  013314  012705  177777                  MOV     #-1,R5          ;SET UP/DOWN SWITCH TO DOWN
4251  013320  000722                          BR      1$              ;GO DO THE NEXT SEEK
4252  013322  005337  004176          4$      DEC     @#DTADPB+12     ;MOVE TO NEXT CYLINDER
4253  013326  023727  004176  000000          CMP     @#DTADPB+12,#0  ;OUT OF CYLINDERS?
4254  013334  003314                          BGT     1$              ;NO--GO DO THE NEXT SEEK
4255  013336  000424                          BR      8$              ;GO TO THE EXIT
4256  013340  042777  000101  166032  7$      BIC     #101,@PKCS      ;STOP THE CLOCK
4257  013346  005037  177776                  CLR     @#PS            ;DROP THE PRIORITY
4258  013352  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
4259  013354  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
4260  013356  104412                          SAVREG                  ;SAVE R0-R5
4261  013360  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4262  013364  004737  042034                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4263  013370  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4264  013376  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4265  013404  104413                          RESREG                  ;RESTORE R0-R5
4266  013406  104020                          ERROR   20              ;REPORT CLOCK OVERFLOW
4267  013410                          8$
4268  013410  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT
4269  013416  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4270  013424  004737  024074                  JSR     PC,@#ST CLK     ;INITIALIZE THE CLOCK
4271  013430  012777  037376  021060          MOV     #ISR,@RPVEC     ;RESTORE RH11/RP04/5/6 INT  VECTOR
4272  013436  004037  027252                  JSR     R0,@#TYPTIM     ;GO TYPE THE TIMES
4273  013442  003014                          T10                     ;POINTER
4274  013444  000004                  EXIT13  SCOPE                   ;LOOP ?
4275
4276
4277                                  ;;**********************************************************************
4278                                  ;*TEST 14        ACCESS TIME MEASURMENT TEST
4279
4280                                  ;*      THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
4281                                  ;*      CYLINDER 'LC', THEN A REVERSEK FROM CYLINDER 'LC' TO
4282                                  ;*      CYLINDER 0.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
4283                                  ;*      ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASURMENT
4284                                  ;*      THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL
4285                                  ;*      OF 256 SEEKS).  THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS
4286                                  ;*      CYLINDER 'LC' DEFAULTS TO 136 (10) FOR AN RP04/5 OR TO 255 (10)
```

```
4287                                        ,*      FOR AN RP06.
4288
4289                                        ;,*************************************************************************
4290  013446                                TST14
4291  013446  000240                               NOP
4292  013450  033737  001454  001234               BIT     @#BITS+<14*2>,TSTNMS ;DO THIS TEST?
4293  013456  001002                               BNE     64$             ;YES--BRANCH
4294  013460  000137  014164                       JMP     TST15           ;NO--GO TO THE NEXT TEST
4295  013464  012737  000014  001102  64$:  MOV     #14,@#$TSTNM    ;SET UP TEST NUMBER AND
4296                                               ;CLEAR THE ERROR FLAG ($ERFLG)
4297  013472  004737  024610                       JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
4298  013476  012737  013446  001110               MOV     #TST14,@#$LPERR ;SETUP THE ERROR LOOP ADDRESS
4299  013504  013777  001102  165430               MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4300  013512  013737  001506  001204               MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
4301  013520  112737  000031  001115               MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
4302  013526  005737  001244                       TST     @#CLKSTA        ;KW11-P CLOCK?
4303  013532  003002                               BGT     1$              ,YES--START TEST
4304  013534  000137  014164                       JMP     TST15           ,NO--GO TO NEXT TEST
4305  013540  012737  013540  001106  1$:  MOV     #1$,$LPADR      ,SET THE LOOP ADDRESS
4306  013546  004037  026670                       JSR     R0,@#SRCH00     ;DO A MASSBUS INIT & RECAL
4307  013552  000402                               BR      2$              ;RETURN HERE IF NO ERROR
4308  013554  000137  014162                       JMP     EXIT14          ;RETURN HERE ON ERROR
4309  013560  012703  003024  2$:.  MOV     #T11,R3         ;PARAMETER POINTER
4310  013564  012737  014162  001206               MOV     #EXIT14,$ESCAPE ;;ESCAPE TO EXIT14 ON ERROR
4311  013572  012706  001100  TEST14:  MOV     #STACK,SP       ;SETUP STACK
4312  013576  012701  000200                       MOV     #128.,R1        ;REPEAT "0-136-0" 128 TIMES
4313  013602  004737  027054                       JSR     PC,@#STRTMR     ;INIT. THE COUNTERS
4314  013606  012777  014056  165560               MOV     #7$,@PKV        ;SET UP VECTOR IN CASE OF OVERFLOW
4315  013614  012777  027052  020674               MOV     #DORTI,@RPVEC   ;SETUP RP04/5/6 VECTOR
4316  013622  005077  165554  1$:  CLR     @PKB            ,START COUNT AT ZERO
4317  013626  013764  001512  000034               MOV     LC,RPCA(R4)     ,'MIDDLE' CYLINDER
4318  013634  012764  000105  000000               MOV     #SEEK,RPCS1(R4) ;START A SEEK
4319  013642  012777  000131  165530               MOV     #131,@PKCS      ;START THE CLOCK
4320  013650  000001                               WAIT                    ,WAIT ON INTERRUPT
4321  013652  042777  000101  165520               BIC     #101,@PKCS      ;STOP CLOCK
4322  013660  032764  040000  000012               BIT     #BIT14,RPDS1(R4) ;ERR=1?
4323  013666  001415                               BEQ     2$              ;NO--BRANCH
4324  013670  104412                               SAVREG                  ;SAVE R0-R5
4325  013672  012702  004164                       MOV     #DTADPB,R2      ,DPB POINTER
4326  013676  004737  042034                       JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
4327  013702  012764  000040  000010               MOV     #BIT05,RPCS2(R4) ,MASSBUS CLEAR
4328  013710  013764  004164  000010               MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4329  013716  104413                               RESREG                  ;RESTORE R0-R5
4330  013720  104017                               ERROR   17
4331  013722  005005                               CLR     R5              ,SET UP/DOWN SWITCH TO UP
4332  013724  004737  027120  2$:  JSR     PC,@#COUNT      ;UPDATE THE COUNT
4333  013730  004737  026446                       JSR     PC,@#TWOMS      ,STALL FOR 2 MILLISECONDS
4334  013734  005077  165442                       CLR     @PKB            ,START THE COUNT AT ZERO
4335  013740  013764  001510  000034               MOV     FC,RPCA(R4)     ;BEGINNING CYLINDER
4336  013746  012764  000105  000000               MOV     #SEEK,RPCS1(R4) ;START A SEEK
4337  013754  012777  000131  165416               MOV     #131,@PKCS      ;START THE CLOCK
4338  013762  000001                               WAIT                    ,WAIT ON INTERRUPT
4339  013764  042777  000101  165406               BIC     #101,@PKCS      ,STOP THE CLOCK
4340  013772  032764  040000  000012               BIT     #BIT14,RPDS1(R4) ,ERR=1?
4341  014000  001415                               BEQ     3$              ,NO--BRANCH
4342  014002  104412                               SAVREG                  ,SAVE R0-R5
```

```
4343  014004  012702  004164                  MOV     #DTADPB,R2          ;DPB POINTER
4344  014010  004737  042034                  JSR     PC,@#SVRH11         ;SAVE ALL THE RH11/RP04 REGISTERS
4345  014014  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4346  014022  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4347  014030  104413                          RESREG                      ;RESTORE R0-R5
4348  014032  104017                          ERROR   17
4349  014034  012705  177777          3$      MOV     #-1,R5              ;SET UP/DOWN SWITCH TO DOWN
4350  014040  004737  027120                  JSR     PC,@#COUNT          ;UPDATE THE COUNT
4351  014044  004737  026446                  JSR     PC,@#TWOMS          ;STALL FOR 2 MILLISECONDS
4352  014050  005301                          DEC     R1                  ;DONE?
4353  014052  003263                          BGT     1$                  ;NO--BRANCH
4354  014054  000424                          BR      8$                  ;YES--EXIT
4355  014056  042777  000101  165314  7$      BIC     #101,@PKCS          ;STOP THE CLOCK
4356  014064  005037  177776                  CLR     @#PS                ;DROP THE PRIORITY
4357  014070  012600                          MOV     (SP)+,R0            ;PC OF WAIT+2
4358  014072  005726                          TST     (SP)+               ,POP THE PS FROM THE STACK
4359  014074  104412                          SAVREG                      ,SAVE R0-R5
4360  014076  012702  004164                  MOV     #DTADPB,R2          ;DPB POINTER
4361  014102  004737  042034                  JSR     PC,@#SVRH11         ;SAVE ALL THE RH11/RP04 REGISTERS
4362  014106  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4363  014114  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4364  014122  104413                          RESREG                      ;RESTORE R0-R5
4365  014124  104020                          ERROR   20                  ,CLOCK OVERFLOWED
4366  014126                          8$
4367  014126  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
4368  014134  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ,SELECT DRIVE
4369  014142  004737  024074                  JSR     PC,@#ST CLK         ;INITIALIZE THE CLOCK
4370  014146  012777  037376  020342          MOV     #ISR,@RPVEC         ,RESTORE RH11/RP04/5/6 INT  VECTOR
4371  014154  004037  027252                  JSR     R0,@#TYPTIM         ,GO TYPE THE TIMES
4372  014160  003024                          T11                         ,POINTER
4373  014162  000004                  EXIT14  SCOPE                       ;LOOP ?
4374
4375                                          ,;*****************************************************************
4376                                          ,*TEST 15       MAXIMUM SEEK TIMING TEST
4377
4378                                          ,*      THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
4379                                          ,*      CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
4380                                          ,*      CYLINDER 0.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE
4381                                          ,*      THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
4382                                          ,*      TIME.  THIS SEQUENCE IS REPEATED 128 TIMES (FOR
4383                                          ,*      A TOTAL OF 256 SEEKS).  THE MAXIMUM SEEK TIME MUST BE LESS THAN
4384                                          ,*      54 MS   'LC' DEFAULTS TO 410 (10) FOR RP04/5'S AND TO 814 (10)
4385                                          ,*      FOR RP06'S
4386
4387                                          ,,*****************************************************************
4388  014164                          TST15
4389  014164  000240                          NOP
4390  014166  033737  001456  001234          BIT     @#BITS+<15*2>,TSTNMS ,DO THIS TEST?
4391  014174  001002                          BNE     64$                 ,YES--BPANCH
4392  014176  000137  014702                  JMP     TST16               ;NO--GO TO THE NEXT TEST
4393  014202  012737  000015  001102  64$     MOV     #15,@#$TSTNM        ,SET UP TEST NUMBER AND
4394                                                                      ;CLEAR THE ERROR FLAG ($ERFLG)
4395  014210  004737  024610                  JSR     PC,LODPRM           ,LOAD THE PARMETERS FOR THE TEST
4396  014214  012737  014164  001110          MOV     #TST15,@#$LPERR ,SETUP THE ERROR LOOP ADDRESS
4397  014222  013777  001102  164712          MOV     $TSTNM,@DISPLAY  ,LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4398  014230  013737  001506  001204          MOV     @#RPT,$TIMES    ,GET THE ITERATION COUNT
```

```
4399  014236  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
4400  014244  005737  001244                  TST     @#CLKSTA        ,KW11-P CLOCK
4401  014250  003002                          BGT     1$              ;YES--START TEST
4402  014252  000137  014702                  JMP     TST16           ,NO--GO TO NEXT TEST
4403  014256  012737  014256  001106  1$:     MOV     #1$,$LPADR      ;SETUP THE LOOP ADDRESS
4404  014264  004037  026670                  JSR     R0,@#SRCH00     ;DO A MASSBUS INIT & RECAL
4405  014270  000402                          BR      2$              ;RETURN HERE IF NO ERROR
4406  014272  000137  014700                  JMP     EXIT15          ;RETURN HERE ON ERROR
4407  014276  012703  003034          2$      MOV     #T12,R3         ;PARAMETER POINTER
4408  014302  012737  014700  001206          MOV     #EXIT15,$ESCAPE ;;ESCAPE TO EXIT15 ON ERROR
4409  014310  012706  001100          TEST15  MOV     #STACK,SP       ;SETUP STACK
4410  014314  012701  000200                  MOV     #128.,R1        ;REPEAT "0-'LC'-0" 128 TIMES
4411  014320  004737  027054                  JSR     PC,@#STRTMR     ;INIT. THE TIMERS
4412  014324  012777  014574  165042          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
4413  014332  012777  027052  020156          MOV     #DORT1,@RPVEC   ;SETUP RP04/5/6 VECTOR
4414  014340  005077  165036          1$      CLR     @PKB            ;START COUNTING FROM ZERO
4415  014344  013764  001512  000034          MOV     LC,RPCA(R4)     ;MAXIMUM CYLINDER
4416  014352  012764  000105  000000          MOV     #SEEK,RPCS1(R4) ;START A SEEK
4417  014360  012777  000131  165012          MOV     #131,@PKCS      ,START THE CLOCK
4418  014366  000001                          WAIT                    ;WAIT ON INTERRUPT
4419  014370  042777  000101  165002          BIC     #101,@PKCS      ;STOP THE CLOCK
4420  014376  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ERR=1?
4421  014404  001415                          BEQ     2$              ;NO--BRANCH
4422  014406  104412                          SAVREG                  ;SAVE R0-R5
4423  014410  012702  004164                  MOV     #DTADPB,R2      ,DPB POINTER
4424  014414  004737  042034                  JSR     PC,@#SVRH11     ,SAVE ALL THE RH11/RP04 REGISTERS
4425  014420  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
4426  014426  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4427  014434  104413                          RESREG                  ,RESTORE R0-R5
4428  014436  104017                          ERROR   17
4429  014440  005005                  2$      CLR     R5              ,SET THE UP/DOWN SWITCH TO UP
4430  014442  004737  027120                  JSR     PC,@#COUNT      ,UP THE COUNT
4431  014446  004737  026446                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
4432  014452  005077  164724                  CLR     @PKB            ,START COUNT AT ZERO
4433  014456  013764  001510  000034          MOV     FC,RPCA(R4)     ,BEGINNING CYLINDER
4434  014464  012764  000105  000000          MOV     #SEEK,RPCS1(R4) ,START A SEEK
4435  014472  012777  000131  164700          MOV     #131,@PKCS      ,START THE CLOCK
4436  014500  000001                          WAIT                    ,WAIT ON INTERRUPT
4437  014502  042777  000101  164670          BIC     #101,@PKCS      ,STOP THE CLOCK
4438  014510  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;"ERR"=1?
4439  014516  001415                          BEQ     3$              ;NO--BRANCH
4440  014520  104412                          SAVREG                  ,SAVE R0-R5
4441  014522  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
4442  014526  004737  042034                  JSR     PC,@#SVRH11     ,SAVE ALL THE RH11/RP04 REGISTERS
4443  014532  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ,MASSBUS CLEAR
4444  014540  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
4445  014546  104413                          RESREG                  ,RESTORE R0-R5
4446  014550  104017                          ERROR   17              ;REPORT THE ERROR
4447  014552  012705  177777          3$      MOV     #-1,R5          ,SET THE UP/DOWN SWITCH TO DOWN
4448  014556  004737  027120                  JSR     PC,@#COUNT      ,UPDATE THE COUNT
4449  014562  004737  026446                  JSR     PC,@#TWOMS      ,STALL FOR 2 MILLISECONDS
4450  014566  005301                          DEC     R1              ,DONE?
4451  014570  003263                          BGT     1$              ,NO--BRANCH
4452  014572  000424                          BR      8$              ,YES--EXIT
4453  014574  042777  000101  164576  7$      BIC     #101,@PKCS      ,STOP THE CLOCK
4454  014602  005037  177776                  CLR     @#PS            ,DROP THE PRIORITY
```

```
 4455  014606  012600                        MOV     (SP)+,R0        ,PC OF WAIT+2
 4456  014610  005726                        TST     (SP)+           ;POP THE PS FROM THE STACK
 4457  014612  104412                        SAVREG                  ,SAVE R0-R5
 4458  014614  012702  004164                MOV     #DTADPB,R2      ;DPB POINTER
 4459  014620  004737  042034                JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
 4460  014624  012764  000040  000010        MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
 4461  014632  013764  004164  000010        MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
 4462  014640  104413                        RESREG                  ;RESTORE R0-R5
 4463  014642  104020                        ERROR   20              ;CLOCK OVERFLOWED
 4464  014644                          8$
 4465  014644  012764  000040  000010        MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
 4466  014652  013764  004164  000010        MOV     @#DTADPB,RPCS2(R4) ,SELECT DRIVE
 4467  014660  004737  024074                JSR     PC,@#ST CLK     ,INITIALIZE THE CLOCK
 4468  014664  012777  037376  017624        MOV     #ISR,@RPVEC     ,RESTORE RH11/RP04/5'6 INT  VECTOR
 4469  014672  004037  027252                JSR     R0,@#TYPTIM     ,GO TYPE THE TIMES
 4470  014676  003034                        T12                     ,POINTER
 4471  014700  000004                  EXIT15  SCOPE                   ,LOOP ?
```

```
4472
4473                                    SBTTL  *** ADDRESSING TESTS ***
4474

        ,/*  /*  /*  /*  /*  /*  /*./*  /*  /*  /*./*:/*:/*./*./*  /*./*
        ,/*  /*  /*  /*  /*  /*  /*./*·/*  /*  /*:/*:/*:/*./*·/*  /*./*
        ,/*  /*  /*  /*  /*  /*./*  /*·/*  /*./*·/*·/*:/*:/*  /*  /*·/*

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

        ,/  */  */  */  */  */  */  */  */  */  */  */  */  */· */· */  */: */  *
         /  */  */  */  */  */  */  */  */  */  */  */  */. */· */: */: */  */  *
        ,'  */  */  */  */  */  */  */  */  */  */  */  */  */  */: */: */· */  *

4488
4489                                    ,,**********************************************************************
4490                                    ,*TEST 16       SECTOR ADDRESSING TEST
4491
4492                                    ,*      THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT"   THE
4493                                    ,*      DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR
4494                                    ,*      BEING WRITTEN.  A WRITE CHECK IS PERFORMED, THE BUFFER IS
4495                                    ,*      CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED.  THEN SECTOR 0
4496                                    ,*      IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED   THEN
4497                                    ,*      SECTOR 1 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED
4498                                    ,*      THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
4499                                    ,*      REWRITE SECTOR 21 AND WRITE CHECK SECTORS 0-21
4500
4501                                    ,,**********************************************************************
4502  014702                            TST16
4503  014702  000240                            NOP
4504  014704  033737  001460  001234            BIT     @#BITS+<16*2>,TSTNMS ;DO THIS TEST?
4505  014712  001002                            BNE     64$             ,YES--BRANCH
4506  014714  000137  015300                    JMP     TST17           ,NO--GO TO THE NEXT TEST
4507  014720  012737  000016  001102  64$       MOV     #16,@#$TSTNM    ,SET UP TEST NUMBER AND
4508                                                                    ,CLEAR THE ERROR FLAG ($ERFLG)
4509  014726  004737  024610                    JSR     PC,LODPRM       ,LOAD THE PARMETERS FOR THE TEST
4510  014732  012737  014776  001110            MOV     #TEST16,@#$LPERR  ,SETUP THE LOOP ON ERROR ADDRESS
4511  014740  013777  001102  164174            MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4512  014746  013737  001506  001204            MOV     @#RPT,$TIMES    ,GET THE ITERATION COUNT
4513  014754  113737  001364  001115            MOVB    ERR.CT,$ERMAX   ,MAX ERRORS ALLOWED FOR TEST
4514  014762  012737  015276  001252            MOV     #EXIT16,@#BYPASS
4515  014770  012737  014776  001106            MOV     #TEST16,$LPADR  ,SETUP THE LOOP ADDRESS
4516  014776  012706  001100          TEST16    MOV     #STACK,SP       ,SET THE STACK POINTER
4517  015002  004737  027704                    JSR     PC,@#FILBUF     ;FILL THE BUFFER WITH SECTOR ADDRESSES
4518  015006  013737  001510  004176            MOV     @#FC,@#DTADPB+12 ,CYLINDER
4519  015014  113737  001516  004175            MOVB    @#FT,@#DTADPB+11 ;TRACK
4520  015022  105037  004174                    CLRB    @#DTADPB+10     ;SECTOR
4521  015026  013737  001352  004170            MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
4522  015034  012737  047712  004172            MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
4523  015042  012737  015042  001110            MOV     # ,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4524  015050  012706  001100                    MOV     #STACK,SP       ,LOAD THE STACK POINTER
4525  015054  012737  000161  004166            MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
4526  015062  004037  025524                    JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4527  015066  012737  000151  004166            MOV     #WRCKD,@#DTADPB+2 ,COMMAND=WRITE CHECK DATA
```

```
4528  015074  012737  015074  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4529  015102  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4530  015106  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4531  015112  012737  015112  001110          MOV     # ,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4532  015120  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4533  015124  004037  027742                  JSR     R0,@#CLRBUF     ;CLEAR BUFFER
4534  015130  012737  000171  004166          MOV     #READ,@#DTADPB+2 ,COMMAND = READ
4535  015136  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4536  015142  004037  030010                  JSR     R0,@#CKSCTR     ;CHECK THE SECTOR DATA READ
4537  015146  012700  047712                  MOV     #BUFFER,R0      ;BUFFER ADDRESS
4538  015152  005001                          CLR     R1              ;FIRST SECTOR
4539  015154  012737  015154  001110          MOV     # ,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4540  015162  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4541  015166  012737  000161  004166  1$      MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
4542  015174  012737  177400  004170          MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
4543  015202  010037  004172                  MOV     R0,@#DTADPB+6   ;BUFFER ADDRESS
4544  015206  110137  004174                  MOVB    R1,@#DTADPB+10  ;SECTOR
4545  015212  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4546  015216  012737  015216  001110          MOV     # ,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4547  015224  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4548  015230  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
4549  015236  013737  001352  004170          MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
4550  015244  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
4551  015252  105037  004174                  CLRB    @#DTADPB+10     ;SECTOR
4552  015256  004037  025524                  JSR     R0,@#DRVCAL     ,START A DATA TRANSFER
4553  015262  062700  001000                  ADD     #512.,R0        ;MOVE TO NEXT SECTOR
4554  015266  005201                          INC     R1
4555  015270  023701  001630                  CMP     PRMLMT+22,R1    ,DONE?
4556  015274  103334                          BHIS    1$              ,NO--BRANCH
4557  015276  000004                  EXIT16  SCOPE                   ,LOOP ?
4558
4559                                  ,,**********************************************************************
4560                                  ,*TEST 17       TRACK ADDRESSING TEST
4561
4562                                  ,*      THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
4563                                  ,*      IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
4564                                  ,*      GETTING ITS OWN TRACK ADDRESS
4565                                  ,*      A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO ENSURE
4566                                  ,*      THE DATA IS VALID   THEN TRACK 0 IS REWRITTEN AND TRACK 1
4567                                  ,*      THROUGH TRACK 18 IS WRITE CHECKED   THEN TRACK 1 IS
4568                                  ,*      REWRITTEN AND TRACK 2 THROUGH TRACK 18 IS WRITE CHECKED
4569                                  ,*      THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING TRACK 17
4570                                  ,*      AND WRITE CHECKING TRACK 18.
4571
4572                                  ,,**********************************************************************
4573  015300                          TST17
4574  015300  000240                          NOP
4575  015302  033737  001462  001234          BIT     @#BITS+<17*2>,TSTNMS ;DO THIS TEST?
4576  015310  001002                          BNE     64$             ,YES--BPANCH
4577  015312  000137  015720                  JMP     TST20           ;NO--GO TO THE NEXT TEST
4578  015316  012737  000017  001102  64$     MOV     #17,@#$TSTNM    ;SET UP TEST NUMBER AND
4579                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
4580  015324  004737  024610                  JSR     PC,LODPRM       ,LOAD THE PARMETERS FOR THE TEST
4581  015330  012737  015374  001110          MOV     #TEST17,@#$LPERR  ,SETUP THE LOOP ON ERROR ADDRESS
4582  015336  013777  001102  163576          MOV     $TSTNM,@DISPLAY  ,LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4583  015344  013737  001506  001204          MOV     @#RPT,$TIMES    ,GET THE ITERATION COUNT
```

```
4584  015352  113737  001364  001115          MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
4585  015360  012737  015716  001252          MOV     #EXIT17,@#BYPASS
4586  015366  012737  015374  001106          MOV     #TEST17,$LPADR  ;SETUP THE LOOP ADDRESS
4587  015374  012706  001100          TEST17. MOV     #STACK,SP       ;SET THE STACK POINTER
4588  015400  004737  027704                  JSR     PC,@#FILBUF     ;FILL THE BUFFER WITH TRACK ADDRESS
4589  015404  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
4590  015412  013737  001510  004176          MOV     @#FC,@#DTADPB+12 ;CYLINDER
4591  015420  113737  001524  004174          MOVB    @#FS,@#DTADPB+10 ;SECTOR
4592  015426  012737  177400  004170          MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
4593  015434  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
4594  015442  005000                          CLR     R0              ;TRACK=0
4595  015444  012737  015444  001110          MOV     # ,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
4596  015452  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4597  015456  110037  004175          1$      MOVB    R0,@#DTADPB+11  ;TRACK ADDRESS
4598  015462  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4599  015466  062737  001000  004172          ADD     #256.*2 ,@#DTADPB+6 ,UPDATE BUFFER ADDRESS
4600  015474  005200                          INC     R0              ,UPDATE TRACK NUMBER
4601  015476  022700  000023                  CMP     #19.,R0         ,OUT OF TRACKS?
4602  015502  003365                          BGT     1$              ,NO--BRANCH
4603  015504  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ,BUFFER ADDRESS
4604  015512  005000                          CLR     R0
4605  015514  012737  015514  001110          MOV     # ,$LPERR        ,SETUP THE ERROR LOOP ADDRESS
4606  015522  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
4607  015526  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK
4608  015534  110037  004175          2$      MOVB    R0,@#DTADPB+11  ;TRACK ADDRESS
4609  015540  004037  025524                  JSR     R0,@#DRVCAL     ,START A DATA TRANSFER
4610  015544  062737  001000  004172          ADD     #256.*2.,@#DTADPB+6 ;UPDATE BUFFER ADDRESS
4611  015552  005200                          INC     R0              ;UPDATE TRACK NUMBER
4612  015554  022700  000023                  CMP     #19 ,R0 ,OUT OF TRACKS?
4613  015560  003365                          BGT     2$              ;NO--BRANCH
4614  015562  005000                          CLR     R0              ,FIRST TRACK ADDRESS
4615  015564  110037  004175          3$      MOVB    R0,@#DTADPB+11  ,TRACK
4616  015570  010001                          MOV     R0,R1           ;FORM BUFFER ADDRESS
4617  015572  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ,BUFFER ADDRESS
4618  015600  005301                  4$      DEC     R1
4619  015602  002411                          BLT     5$
4620  015604  062737  001000  004172          ADD     #256 *2 ,@#DTADPB+6
4621  015612  000772                          BR      4$
4622  015614  012737  015614  001110          MOV     #.,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
4623  015622  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4624  015626  012737  000161  004166  5$      MOV     #WRITE,@#DTADPB+2 ,COMMAND=WRITE DATA
4625  015634  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4626  015640  062737  001000  004172  6$      ADD     #256.*2 ,@#DTADPB+6 ;UPDATE BUFFER ADDRESS
4627  015646  105237  004175                  INCB    @#DTADPB+11     ;MOVE TO NEXT TRACK
4628  015652  012737  015652  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4629  015660  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
4630  015664  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
4631  015672  004037  025524                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
4632  015676  122737  000022  004175          CMPB    #18.,@#DTADPB+11 ;OUT OF TRACKS?
4633  015704  003355                          BGT     6$              ,NO--BRANCH
4634  015706  005200                          INC     R0              ;NEXT TRACK TO WRITE
4635  015710  022700  000022                  CMP     #18 ,R0         ,OUT OF TRACKS?
4636  015714  003323                          BGT     3$              ,NO--BRANCH
4637  015716  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
```

```
                                        ;*      133333  004000  173777  155555  177757  066667  177777  000000
                                        ;*      165555  010000  167777  172666  177767  153333  177777  000000
                                        ;*      133333  020000  157777  155555  177773  066667  177777  000000
                                        ;*      165555  040000  137777  172666  177775  153333  177777  000000
                                        ;*      133333  100000  077777  155555  177776  066667  177777  000000
                                        ,*

                                        ;,**********************************************************************
015720                                  TST20:
015720  000240                                  NOP
015722  033737  001424  001236                  BIT     BITS+<20*2-40>,TSTNMS+2  ;DO THIS TEST ?
015730  001002                                  BNE     64$                     ;YES--BRANCH
015732  000137  016440                          JMP     TST21                   ;NO--GO TO THE NEXT TEST
015736  012737  000020  001102  64$             MOV     #20,@#$TSTNM            ;SET UP TEST NUMBER AND
                                                                                ;CLEAR THE ERROR FLAG ($ERFLG)
015744  004737  024610                          JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
015750  012737  016122  001110                  MOV     #TEST20,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
015756  013777  001102  163156                  MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
015764  013737  001506  001204                  MOV     @#RPT,$TIMES            ;GET THE ITERATION COUNT
015772  113737  001364  001115                  MOVB    ERR.CT,$ERMAX           ;MAX ERRORS ALLOWED FOR TEST
016000  012737  016000  001106                  MOV     # ,$LPADR               ;SETUP THE LOOP ADDRESS
016006  005000                                  CLR     R0                      ,CLEAR SWITCH
016010  005004                                  CLR     R4                      ,FORM WORD COUNT IN R4
016012  013701  001526                          MOV     @#LS,R1
016016  163701  001524                          SUB     @#FS,R1
016022  002004                                  BGE     1$                      ;BRANCH IF FS < OR = LS
016024  063701  001630                          ADD     PRMLMT+22,R1            ,ADD MAXIMUM SECTOR ADDRESS TO
016030  005201                                  INC     R1                      ,MAKE THE DIFFERENCE POSITIVE
016032  005100                                  COM     R0                      ,SET SWITCH
016034  062704  000400          1$              ADD     #256 ,R4
016040  005301                                  DEC     R1
016042  002374                                  BGE     1$
016044  005404                                  NEG     R4
016046  010405                                  MOV     R4,R5                   ,COPY NORMAL WORD COUNT INTO SMALL WC
016050  005700                                  TST     R0                      ;SWITCH SET?
016052  001412                                  BEQ     3$                      ,NO--BRANCH
016054  005005                                  CLR     R5                      ;FORM WORD COUNT FOR LS < FS
016056  013701  001630                          MOV     PRMLMT+22,R1
016062  163701  001524                          SUB     @#FS,R1
016066  062705  000400          2$              ADD     #256 ,R5
016072  005301                                  DEC     R1
016074  002374                                  BGE     2$
016076  005405                                  NEG     R5
016100  113737  001524  004174  3$              MOVB    @#FS,@#DTADPB+10 ,SECTOR
016106  012737  047712  004172                  MOV     #BUFFER,@#DTADPB+6 ;DATA BUFFER
016114  012737  016436  001252                  MOV     #EXIT20,@#BYPASS
016122  012706  001100          TEST20          MOV     #STACK,SP               ;LOAD THE STACK POINTER
016126  005037  001334                          CLR     @#WCEFLG                ;CLEAR THE WRITE CHECK ERROR FLAG
016132  013701  001510                          MOV     @#FC,R1 ;PICKUP FIRST CYLINDER
016136  000407                                  BR      2$
016140  005720          1$                      TST     (R0)+                   ;MOVE TO NEXT DATA PATTERN
016142  022700  000040                          CMP     #16 *2 ,R0              ,OUT OF PATTERNS?
016146  003004                                  BGT     3$                      ,NO--BRANCH
016150  004037  027630                          JSR     R0,@#INCCYL             ,MOVE TO NEXT CYLINDER
016154  000530                                  BR      EXIT20                  ,OUT OF CYLINDERS
016156  005000          2$                      CLR     R0                      ,START WITH PATTERN 0
```

```
 4750  016160  036037  001424  001530  3$:     BIT     BITS(R0),@#PAT   ;THIS PATTERN SELECTED?
 4751  016166  001764                          BEQ     1$               ;NO--BRANCH
 4752  016170  013702  001516                  MOV     @#FT,R2          ;FIRST TRACK
 4753  016174  010137  004176                  MOV     R1,@#DTADPB+12   ;CYLINDER
 4754  016200  110237  004175          4$:     MOVB    R2,@#DTADPB+11   ;TRACK
 4755  016204  010437  004170                  MOV     R4,@#DTADPB+4    ;WORD COUNT
 4756  016210  023701  001512                  CMP     LC,R1            ;LAST DISK CYLINDER?
 4757  016214  003005                          BGT     5$               ;NO--BRANCH
 4758  016216  022702  000022                  CMP     #18.,R2          ;LAST DISK TRACK?
 4759  016222  003002                          BGT     5$               ;NO--BRANCH
 4760  016224  010537  004170                  MOV     R5,@#DTADPB+4    ;SHORT WORD COUNT
 4761  016230  017703  162704          5$      MOV     @SWR,R3          ;INHIBIT WRITE AND
 4762  016234  005103                          COM     R3               ;WRITE CHECK?
 4763  016236  032703  000030                  BIT     #SW04!SW03,R3
 4764  016242  001436                          BEQ     7$               ;YES--BRANCH
 4765  016244  004737  030360                  JSR     PC,@#SETBUF      ;MOVE DATA PATTERN INTO THE BUFFER
 4766  016250  032777  000020  162662          BIT     #SW04,@SWR       ;INHIBIT WRITE?
 4767  016256  001012                          BNE     6$               ;YES--BRANCH
 4768  016260  012737  016260  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4769  016266  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4770  016272  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
 4771  016300  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4772  016304  032777  000010  162626  6$      BIT     #SW03,@SWR       ;INHIBIT WRITE CHECK?
 4773  016312  001012                          BNE     7$               ;YES--BRANCH
 4774  016314  012737  016314  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4775  016322  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4776  016326  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
 4777  016334  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4778  016340  005737  001334          7$      TST     @#WCEFLG         ;WRITE CHECK ERROR FLAG SET?
 4779  016344  001404                          BEQ     8$               ;NO--BRANCH
 4780  016346  032777  000001  162564          BIT     #SW00,@SWR       ;PERFORM READ AFTER FATAL "WCE"?
 4781  016354  001424                          BEQ     9$               ;NO--BRANCH
 4782  016356  032777  000004  162554  8$      BIT     #SW02,@SWR       ;INHIBIT READ DATA AND SOFTWARE COMPARE?
 4783  016364  001020                          BNE     9$               ;YES--BRANCH
 4784  016366  012737  016366  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4785  016374  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4786  016400  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ
 4787  016406  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4788  016412  032777  000002  162520          BIT     #SW01,@SWR       ;COMPARE THE DATA?
 4789  016420  001002                          BNE     9$               ;NO--BRANCH
 4790  016422  004737  030450                  JSR     PC,@#DATCMP      ;YES--DO IT
 4791  016426  004037  027600          9$      JSR     R0,@#INCTRK      ;MOVE TO NEXT TRACK
 4792  016432  000642                          BR      1$               ;OUT OF TRACKS GO TO NEXT PATTERN
 4793  016434  000661                          BR      4$               ;LOOP
 4794  016436  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 ADDRRESS
                                ;*      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

                                ;;**********************************************************************
016440                          TST21
016440  000240                          NOP
016442  033737  001426  001236          BIT     BITS+<21*2-40>,TSTNMS+2  ;DO THIS TEST ?
016450  001002                          BNE     64$             ,YES--BRANCH
016452  000137  017216                  JMP     TST22           ,NO--GO TO THE NEXT TEST
016456  012737  000021  001102  64$:    MOV     #21,@#$TSTNM    ,SET UP TEST NUMBER AND
                                                                ,CLEAR THE ERROR FLAG ($ERFLG)
016464  004737  024610                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
016470  012737  016710  001110          MOV     #TEST21,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
016476  013777  001102  162436          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
016504  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
016512  113737  001364  001115          MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
016520  012737  016520  001106          MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
016526  012737  017214  001252          MOV     #EXIT21,@#BYPASS
016534  012737  176543  023606          MOV     #176543,@#$HINUM ;PRIME THE RANDOM NUMBER GENERATOR
016542  012737  123456  023610          MOV     #123456,@#$LONUM
016550  013737  001510  004176          MOV     @#FC,@#DTADPB+12 ;CYLINDER
016556  013737  001352  004170          MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
016564  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
016572  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMMAND
016600  032737  100000  001220          BIT     #SW15,@#C.SWR   ;WRITE THE DISK PACK BEFORE TESTING?
016606  001027                          BNE     3$              ;NO--BRANCH
016610  004037  030766                  JSR     R0,@#FILRAN     ,FILL DATA BUFFER WITH RANDOM DATA
016614  005037  004174          1$      CLR     @#DTADPB+10     ;SECTOR AND TRACK
016620  012737  016620  001110          MOV     # ,$LPERR       ,SETUP THE ERROR LOOP ADDRESS
016626  012706  001100                  MOV     #STACK,SP       ,LOAD THE STACK POINTER
016632                          2$
016632  004037  025524                  JSR     R0,@#DRVCAL     ,START A DATA TRANSFER
016636  105237  004175                  INCB    @#DTADPB+11     ,NEXT TRACK
```

```
 4851  016642  122737  000023  004175          CMPB    #19.,@#DTADPB+11 ;TIME FOR NEXT CYLINDER
 4852  016650  003370                          BGT     2$               ;NO--BRANCH
 4853  016652  005237  004176                  INC     @#DTADPB+12
 4854  016656  023737  001512  004176          CMP     @#LC,@#DTADPB+12 ;OUT OF CYLINDERS?
 4855  016664  002353                          BGE     1$               ;NO--BRANCH
 4856  016666  012737  177400  004170  3$:     MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
 4857  016674  012737  016710  001106          MOV     #TEST21,@#$LPADR
 4858  016702  012737  016710  001110          MOV     #TEST21,@#$LPERR
 4859  016710  012706  001100          TEST21: MOV     #STACK,SP        ;SET STACK POINTER
 4860  016714  004037  031242                  JSR     R0,@#RANADR      ;GENERATE A RANDOM ADDRESS
 4861  016720  013737  004174  001340          MOV     @#DTADPB+10,@#SVADR ;SAVE THE TRACK/SECTOR
 4862  016726  013737  004176  001342          MOV     @#DTADPB+12,@#SVADR+2 ;SAVE THE CYLINDER
 4863  016734  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
 4864  016742  012701  047712                  MOV     #BUFFER,R1       ;BUFFER ADDRESS
 4865  016746  010137  004172                  MOV     R1,@#DTADPB+6
 4866  016752  004037  031206                  JSR     R0,@#RANPAT      ;GENERATE RANDOM PATTERN
 4867  016756  012737  016756  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4868  016764  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4869  016770  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4870  016774  004037  031242                  JSR     R0,@#RANADR
 4871  017000  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ DATA
 4872  017006  012737  050712  004172          MOV     #BUFFER+512.,@#DTADPB+6 ;BUFFER ADDRESS
 4873  017014  012737  017014  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4874  017022  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4875  017026  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4876  017032  004037  031010                  JSR     R0,@#RANCK       ;CHECK THE DATA
 4877  017036  013737  001340  004174          MOV     @#SVADR,@#DTADPB+10 ;GET ADDRESS OF WHERE THE LAST
 4878  017044  013737  001342  004176          MOV     @#SVADR+2,@#DTADPB+12 ;WRITE WAS PERFORMED
 4879  017052  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
 4880  017060  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
 4881  017066  012737  017066  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4882  017074  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4883  017100  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4884  017104  004037  031242                  JSR     R0,@#RANADR      ;GENERATE A RANDOM ADDRESS
 4885  017110  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ
 4886  017116  012737  050712  004172          MOV     #BUFFER+512.,@#DTADPB+6 ;DATA BUFFER ADDRESS
 4887  017124  012737  017124  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4888  017132  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4889  017136  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4890  017142  004037  031010                  JSR     R0,@#RANCK       ;CHECK THE DATA
 4891  017146  013737  001340  004174          MOV     @#SVADR,@#DTADPB+10 ;GET DISK ADDRESS OF THE
 4892  017154  013737  001342  004176          MOV     @#SVADR+2,@#DTADPB+12 ;LAST WRITE
 4893  017162  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
 4894  017170  012737  047712  004172          MOV     #BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
 4895  017176  012737  017176  001110          MOV     #.,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
 4896  017204  012706  001100                  MOV     #STACK,SP        ;LOAD THE STACK POINTER
 4897  017210  004037  025524                  JSR     R0,@#DRVCAL      ;START A DATA TRANSFER
 4898  017214  000004                  EXIT21. SCOPE                    ;LOOP ?
 4899
 4900                                   .SBTTL  *** RP04 ACCESS TIME ADJUSTMENT TEST ***
 4901
 4902                                   ;+*********************************************************************
 4903                                   ;*TEST 22       RP04 ACCESS TIME ADJUSTMENT TEST
 4904
 4905                                   ;*      THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 136 TO ALLOW THE
 4906                                   ;*      OPERATOR TO ADJUST THE ACCESS TIME ON AN RP04 USING THE
```

```
4907                                    ;*      DDU.  THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
4908                                    ;*      SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.
4909
4910                                    ;;**********************************************************************
4911  017216                            TST22
4912  017216  000240                            NOP
4913  017220  033737  001430  001236            BIT     BITS+<22*2-40>,TSTNMS+2  ;DO THIS TEST ?
4914  017226  001002                            BNE     64$                ;YES--BRANCH
4915  017230  000137  017366                    JMP     $EOP               ;NO--GO TO THE END OF THE PROGRAM
4916  017234  012737  000022  001102  64$:     MOV     #22,@#$TSTNM       ;SET UP TEST NUMBER AND
4917                                            ;CLEAR THE ERROR FLAG ($ERFLG)
4918  017242  004737  024610                    JSR     PC,LODPRM          ;LOAD THE PARMETERS FOR THE TEST
4919  017246  012737  017304  001110            MOV     #TEST22,@#$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
4920  017254  013777  001102  161660            MOV     $TSTNM,@DISPLAY    ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4921  017262  013737  001506  001204            MOV     @#RPT,$TIMES       ;GET THE ITERATION COUNT
4922  017270  112737  000144  001115            MOVB    #100.,$ERMAX       ;MAX ERRORS ALLOWED FOR TEST
4923  017276  012737  017304  001106            MOV     #TEST22,$LPADR     ;SETUP THE LOOP ADDRESS
4924  017304  012706  001100          TEST22:   MOV     #STACK,SP          ;SETUP THE STACK POINTER
4925  017310  013737  001512  004116            MOV     LC,DPB+12          ;ENDING CYLINDER
4926  017316  112737  000105  004106            MOVB    #SEEK,@#DPB+2      ;SEEK=COMMAND
4927  017324  004037  025032                    JSR     R0,@#CALL.A        ;GO EXECUTE THE COMMAND
4928  017330  004037  026364                    JSR     R0,STALL           ;STALL
4929  017334  001360                            .WORD   STALL3             ;ADDRESS OF STALL VALUE
4930  017336  013737  001510  004116            MOV     FC,DPB.A+12        ;STARTING CYLINDER
4931  017344  112737  000105  004106            MOVB    #SEEK,@#DPB+2      ;SEEK=COMMAND
4932  017352  004037  025032                    JSR     R0,@#CALL.A        ;GO EXECUTE THE COMMAND
4933  017356  004037  026364                    JSR     R0,STALL           ;STALL
4934  017362  001360                             WORD   STALL3             ;ADDRESS OF STALL VALUE
4935  017364  000004                    EXIT22: SCOPE                      ;LOOP ?
4936
4937
4938
4939                                     SBTTL  END OF PASS ROUTINE
4940
4941                                    ;;**********************************************************************
4942                                    ;*INCREMENT THE PASS NUMBER ($PASS)
4943                                    ;*INDICATE END-OF-PROGRAM AFTER 8  PASSES THRU THE PROGRAM
4944                                    ;*IF THERES A MONITOR GO TO IT
4945                                    ;*IF THERE ISN'T JUMP TO RESTART
4946
4947  017366                            $EOP
4948  017366  104401  017374                    TYPE    ,65$               ;;TYPE ASCIZ STRING
4949  017372  000410                            BR      64$                ;;GET OVER THE ASCIZ
4950                                    ;;65$    ASCIZ   <CR><LF><LF>/END OF PASS/
4951  017414                            64$:
4952  017414  005737  001232                    TST     @#DRVSEL           ;ANY DRIVES SELECTED?
4953  017420  001434                            BEQ     1$                 ;NO--BRANCH
4954  017422  104401  017430                    TYPE    ,67$               ;;TYPE ASCIZ STRING
4955  017426  000405                            BR      66$                ;;GET OVER THE ASCIZ
4956                                    ;;67$    .ASCIZ  / ON DRIVE/
4957  017442                            66$:
4958  017442  013746  001254                    MOV     @#CHKDRV,-(SP)     ;;SAVE @#CHKDRV FOR TYPEOUT
4959  017446  104403                            TYPOS                      ;;GO TYPE--OCTAL ASCII
4960  017450     002                             BYTE   2                  ;;TYPE 2 DIGIT(S)
4961  017451     000                             BYTE   0                  ;;SUPPRESS LEADING ZEROS
4962  017452  104401  017460                    TYPE    ,69$               ;;TYPE ASCIZ STRING
```

```
 4963  017456  000412                          BR      68$             ;;GET OVER THE ASCIZ
 4964                                ,;69$     .ASCIZ  /   ERRORS DETECTED=/
 4965  017504                        68$
 4966  017504  013746  001112                  MOV     @#$ERTTL,-(SP)  ;;SAVE @#$ERTTL FOR TYPEOUT
 4967  017510  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4968  017512  005037  001112          1$      CLR     @#$ERTTL        ;ZERO ERROR TOTAL
 4969  017516  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 4970  017522  005037  001204                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 4971  017526  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 4972  017532  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG  NUMBER
 4973  017540  005327                          DEC     (PC)+           ;,LOOP?
 4974  017542  000010                $EOPCT    .WORD   8
 4975  017544  003027                          BGT     $DOAGN          ,;YES
 4976  017546  012737                          MOV     (PC)+,@(PC)+    ,;RESTORE COUNTER
 4977  017550  000010                $ENDCT    .WORD   8
 4978  017552  017542                          $EOPCT
 4979  017554  104401  017562                  TYPE    ,65$            ,,TYPE ASCIZ STRING
 4980  017560  000407                          BR      64$             ,,GET OVER THE ASCIZ
 4981                                ,,65$      ASCIZ  <CR><LF>/END OF TEST/
 4982  017600                        64$
 4983  017600  104401  017630                  TYPE    ,$ENULL         ,TYPE NULL CHARACTER
 4984  017604  013700  000042        $GET42    MOV     @#42,R0         ;,GET MONITOR ADDRESS
 4985  017610  001405                          BEQ     $DOAGN          ,,BRANCH IF NO MONITOR
 4986  017612  000005                          RESET                   ,,CLEAR THE WORLD
 4987  017614  004710                $ENDAD    JSR     PC,(R0)         ;;GO TO MONITOR
 4988  017616  000240                          NOP                     ,,SAVE ROOM
 4989  017620  000240                          NOP                     ,,FOR
 4990  017622  000240                          NOP                     ,,ACT11
 4991  017624                        $DOAGN
 4992  017624  000137                          JMP     @(PC)+          ,,RETURN
 4993  017626  006102                $RTNAD     WORD   RESTART
 4994  017630     377     377   000  $ENULL    .BYTE   -1,-1,0         ,,NULL CHARACTER STRING
 4995          017634                          EVEN
```

```
 4996
 4997                                    SBTTL  *** SYSMAC SUBROUTINES ***
 4998
 4999                                    SBTTL  ERROR HANDLER ROUTINE
 5000
 5001                                    ,;**********************************************************************
 5002                                    ,*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 5003                                    ,*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 5004                                    ,*AND GO TO TYPERR ON ERROR
 5005                                    ,*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE
 5006                                    ,*SW15=1        HALT ON ERROR
 5007                                    ,*SW13=1        INHIBIT ERROR TYPEOUTS
 5008                                    ;*SW10=1        BELL ON ERROR
 5009                                    ,*SW09=1        LOOP ON ERROR
 5010                                    ,*CALL
 5011                                    ,*      ERROR   N       ,,ERROR=EMT AND N=ERROR ITEM NUMBER
 5012
 5013  017634                            $ERROR:
 5014  017634  104407                            CKSWR                   ,,TEST FOR CHANGE IN SOFT-SWR
 5015  017636  032777  000400  161274            BIT     #SW08,@SWR      ;SEND ERROR MESSAGE TO TTY?
 5016  017644  001411                            BEQ     7$              ;YES--BRANCH
 5017  017646  005737  001230                    TST     @#LPTAVL        ,IS THERE A LINE PRINTER AVAILABLE?
 5018  017652  001406                            BEQ     7$              ,NO--BRANCH
 5019  017654  013737  001420  001150            MOV     @#LPS,@#$TPS    ,YES--SETUP STATUS
 5020  017662  013737  001422  001152            MOV     @#LPB,@#$TPB    ,AND BUFFER REG 'S FOR LINE PRINTER
 5021  017670  105237  001103            7$:     INCB    $ERFLG          ;,SET THE ERROR FLAG
 5022  017674  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 5023  017676  013777  001102  161236            MOV     $TSTNM,@DISPLAY ,,DISPLAY TEST NUMBER AND ERROR FLAG
 5024  017704  032777  002000  161226            BIT     #BIT10,@SWR     ,,BELL ON ERROR?
 5025  017712  001402                            BEQ     1$              ,,NO - SKIP
 5026  017714  104401  001210                    TYPE    ,$BELL          ,,RING BELL
 5027  017720  005237  001112            1$:     INC     $ERTTL          ;,COUNT THE NUMBER OF ERRORS
 5028  017724  011637  001116                    MOV     (SP),$ERRPC     ,,GET ADDRESS OF ERROR INSTRUCTION
 5029  017730  162737  000002  001116            SUB     #2,$ERRPC
 5030  017736  117737  161154  001114            MOVB    @$ERRPC,$ITEMB  ,,STRIP AND SAVE THE ERROR ITEM CODE
 5031  017744  032777  020000  161166            BIT     #BIT13,@SWR     ,,SKIP TYPEOUT IF SET
 5032  017752  001004                            BNE     20$             ,,SKIP TYPEOUTS
 5033  017754  004737  020054                    JSR     PC,TYPERR       ,,GO TO USER ERROR ROUTINE
 5034  017760  104401  001215                    TYPE    ,$CRLF
 5035  017764                            20$:
 5036  017764  005777  161150            2$:     TST     @SWR            ,,HALT ON ERROR
 5037  017770  100002                            BPL     3$              ,,SKIP IF CONTINUE
 5038  017772  000000                            HALT                    ,,HALT ON ERROR!
 5039  017774  104407                            CKSWR                   ,,TEST FOR CHANGE IN SOFT-SWR
 5040  017776  032777  001000  161134    3$:     BIT     #BIT09,@SWR     ,;LOOP ON ERROR SWITCH SET?
 5041  020004  001402                            BEQ     4$              ,,BR IF NO
 5042  020006  013716  001110                    MOV     $LPERR,(SP)     ,,FUDGE RETURN FOR LOOPING
 5043  020012  005737  001206            4$:     TST     $ESCAPE         ,,CHECK FOR AN ESCAPE ADDRESS
 5044  020016  001402                            BEQ     5$              ,,BR IF NONE
 5045  020020  013716  001206                    MOV     $ESCAPE,(SP)    ,,FUDGE RETURN ADDRESS FOR ESCAPE
 5046  020024                            5$:
 5047  020024  023737  000042  000046            CMP     @#42,@#46               ,ACT11 AUTOMATIC MODE?
 5048  020032  001001                            BNE     6$              ,NO, CONTINUE
 5049  020034  000000                            HALT                    ,HALT ON ERROR
 5050  020036  013737  001414  001150    6$:     MOV     @#TPS,@#$TPS    ,SET STATUS AND BUFFER REG 'S
 5051  020044  013737  001416  001152            MOV     @#TPB,@#$TPB    ,FOR TTY
```

```
5052  020052  000002                          RTI                     ;RETURN FROM ERROR CALL
5053
5054                                  ,;*****************************************************************
5055                                   SBTTL  TYPERR - TYPE ERROR ROUTINE
5056                                  ,THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
5057                                  ,WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
5058                                  ,TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
5059                                  ,CONCERNING THE ERROR.
5060                                  ,CALL
5061                                  ,       JSR     PC,@#TYPERR
5062                                  ,       RETURN
5063
5064  020054  113737  001102  001176  TYPERR. MOVB    @#$TSTNM,@#$TMPO ;SAVE THE TEST NUMBER
5065  020062  104412                          SAVREG                  ;SAVE R0 - R5
5066  020064  162700  000004                  SUB     #4,R0           ;FORM TEST PC
5067  020070  010037  001162                  MOV     R0,@#$REG0      ;COPY R0-R5 IN $REG0-$REG5
5068  020074  010137  001164                  MOV     R1,@#$REG1
5069  020100  010237  001166                  MOV     R2,@#$REG2
5070  020104  010337  001170                  MOV     R3,@#$REG3
5071  020110  010437  001172                  MOV     R4,@#$REG4
5072  020114  010537  001174                  MOV     R5,@#$REG5
5073  020120  113700  001114                  MOVB    @#$ITEMB,R0     ,PICKUP ERROR ITEM NUMBER
5074  020124  010001                          MOV     R0,R1           ,AND COPY IT INTO R1
5075  020126  005300                          DEC     R0              ,FORM INDEX FOR ERROR TABLE
5076  020130  106300                          ASLB    R0
5077  020132  106300                          ASLB    R0
5078  020134  106300                          ASLB    R0
5079  020136  103002                          BCC     1$              ,IS ERROR > 37?
5080  020140  062700  000240                  ADD     #ITEM41-$ERRTB,R0 ,YES--FORM OFFSET
5081  020144  062700  004306          1$      ADD     #$ERRTB,R0      ,FORM ADDRESS
5082  020150  012037  020164                  MOV     (R0)+,2$        ,GET ERROR MESSAGE (EM) POINTER
5083  020154  001447                          BEQ     7$              ;BRANCH IF THERE ISN'T ONE
5084  020156  104401  001215                  TYPE    ,$CRLF          ,"CARRIAGE RETURN - LINE FEED
5085  020162  104401                          TYPE
5086  020164  000000          2$       WORD    0                       ,"EM" POINTER GOES HERE
5087  020166  162701  000041                  SUB     #41,R1          ,SPECIAL ERROR ITEM NUMBER?
5088  020172  100440                          BMI     7$              ;NO--BRANCH
5089  020174  013701  001260                  MOV     @#SVSTAT,R1     ,GET STATUS/ERROR INDICATOR
5090  020200  106301                          ASLB    R1              ,STRIP "DONE" BIT (BIT07)
5091  020202  006301                          ASL     R1              ,STRIP "ERROR" BIT (BIT15)
5092  020204  012702  004254                  MOV     #STATBL,R2      ,1ST ADDRESS ON STATUS MESSAGE POINTERS
5093  020210  005003                          CLR     R3              ,CARRIAGE RETURN-LINE FEED SWITCH
5094  020212  104401  020220                  TYPE    ,65$            ,,TYPE ASCIZ STRING
5095  020216  000402                          BR      64$             ,,GET OVER THE ASCIZ
5096                                ,,65$     ASCIZ   / (/
5097  020224                          64$
5098  020224  012237  020246          3$      MOV     (R2)+,5$        ,MESSAGE POINTER
5099  020230  006301                          ASL     R1              ,TYPE THIS MESSAGE?
5100  020232  103013                          BCC     6$              ,NO--BRANCH
5101  020234  005103                          COM     R3              ,YES--TYFE A "CR" & "LF"?
5102  020236  001002                          BNE     4$              ,NO--BRANCH
5103  020240  104401  001215                  TYPE    ,$CRLF          ,YES
5104  020244  104401                  4$      TYPE
5105  020246  000000          5$      .WORD   0                       ,MESSAGE POINTER GOES HERE
5106  020250  005701                          TST     R1              ;MORE TO TYPE?
5107  020252  001403                          BEQ     6$              ,NO--BRANCH
```

```
 5108  020254  104401  044040                TYPE    ,MSG.SP          ;YES--SPACES
 5109  020260  000761                        BR      3$               ,LOOP
 5110  020262  001360                6$      BNE     3$               ;BRANCH IF NOT FINISHED
 5111  020264  104401  020272                TYPE    ,67$             ,,TYPE ASCIZ STRING
 5112  020270  000401                        BR      66$              ;,GET OVER THE ASCIZ
 5113                                  ,,67$   .ASCIZ  /)/
 5114  020274                        66$
 5115  020274  012037  020310        7$      MOV     (R0)+,8$         ;PICK UP DATA HEADER (DH) POINTER
 5116  020300  001404                        BEQ     9$               ,BRANCH IF NONE
 5117  020302  104401  001215                TYPE    ,$CRLF           ;CARRIAGE RETURN-LINE FEED
 5118  020306  104401                        TYPE
 5119  020310  000000                8$      .WORD   0                ,"DH" POINTER GOES HERE
 5120  020312  012001                9$      MOV     (R0)+,R1         ,PICKUP DATA TABLE (DT) POINTER
 5121  020314  001450                        BEQ     20$              ,BRANCH IF NONE
 5122  020316  005005                        CLR     R5               ,SET INDENT SWITCH
 5123  020320  012000                        MOV     (R0)+,R0         ;DATA FORMAT (DF) POINTER
 5124  020322  012002                        MOV     (R0)+,R2         ,NUMBER OF DH'S TO TYPE
 5125  020324  001441                        BEQ     17$              ,BRANCH IF DH NUMBER IS 0
 5126  020326  005105                        COM     R5               ,NO INDENT
 5127  020330  104401  001215                TYPE    ,$CRLF           ,CARRIAGE RETURN-LINE FEED
 5128  020334  112003                10$     MOVB    (R0)+,R3         ,NUMBER OF DATA WORDS TO TYPE
 5129  020336  112004                        MOVB    (R0)+,R4         ,AND HOW TO TYPE THEM
 5130  020340  006004                11$     ROR     R4               ,OCTAL OR DECIMAL?
 5131  020342  103403                        BCS     12$              ,DECIMAL--BRANCH
 5132  020344  013146                        MOV     @(R1)+,-(SP)     ,,SAVE @(R1)+ FOR TYPEOUT
 5133  020346  104402                        TYPOC                    ,,GO TYPE--OCTAL ASCII(ALL DIGITS)
 5134  020350  000402                        BR      13$
 5135  020352                        12$
 5136  020352  013146                        MOV     @(R1)+,-(SP)     ,,SAVE @(R1)+ FOR TYPEOUT
 5137  020354  104405                        TYPDS                    ,,GO TYPE--DECIMAL ASCII WITH SIGN
 5138  020356  005303                13$     DEC     R3               ,MORE NUMBERS TO TYPE?
 5139  020360  001403                        BEQ     14$              ,NO--BRANCH
 5140  020362  104401  044040                TYPE    ,MSG SP          ,YES--TYPE SEPERATORS
 5141  020366  000764                        BR      11$              ,LOOP
 5142  020370  005302                14$     DEC     R2               ;MORE DH'S?
 5143  020372  003421                        BLE     20$              ,NO--BRANCH
 5144  020374  104401  001215                TYPE    ,$CRLF           ,YES--START A NEW LINE
 5145  020400  005105                        COM     R5               ,INDENT?
 5146  020402  001002                        BNE     15$              ,NO--BRANCH
 5147  020404  104401  044040                TYPE    ,MSG SP          ;YES--TYPE SPACES
 5148  020410  012037  020416        15$     MOV     (R0)+,16$        ,GET NEXT DH
 5149  020414  104401                        TYPE                     ,AND TYPE IT
 5150  020416  000000                16$      WORD   0                ,DH POINTER GOES HERE
 5151  020420  104401  001215                TYPE    ,$CRLF           ,CARRIAGE RETURN-LINE FEED
 5152  020424  005705                        TST     R5               ,INDENT?
 5153  020426  001342                        BNE     10$              ,NO--BRANCH
 5154  020430  104401  044040        17$     TYPE    ,MSG SP          ,YES--TYPE SPACES
 5155  020434  000737                        BR      10$              ,LOOP
 5156  020436  104413                20$     RESREG                   ,RESTORE R0 - R5
 5157  020440  000207                        RTS     PC               ,RETURN
 5158
 5159                                   SBTTL  TYPE ROUTINE
 5160
 5161                                  ,,*********************************************************************
 5162                                  ,*ROUTINE TO TYPE ASCIZ MESSAGE  MESSAGE MUST TERMINATE WITH A 0 BYTE
 5163                                  ,*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED
```

```
5164                                    ;*NOTE1         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER
5165                                    ;*NOTE2         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
5166                                    ;*NOTE3         $FILLC CONTAINS THE CHARACTER TO FILL AFTER
5167                                    ;*
5168                                    ;*CALL
5169                                    ;*1) USING A TRAP INSTRUCTION
5170                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
5171                                    ;*
5172                                    ;*OR
5173                                    ;*      TYPE
5174                                    ;*      MESADR
5175                                    ;*
5176  020442  105737  001157            $TYPE   TSTB    $TPFLG          ;;IS THERE A TERMINAL?
5177  020446  100002                            BPL     1$              ;;BR IF YES
5178  020450  000000                            HALT                    ;;HALT HERE IF NO TERMINAL
5179  020452  000407                            BR      3$              ;;LEAVE
5180  020454  010046                    1$      MOV     R0,-(SP)        ;;SAVE R0
5181  020456  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
5182  020462  112046                    2$      MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
5183  020464  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
5184  020466  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
5185  020470  012600                    60$     MOV     (SP)+,R0        ;;RESTORE R0
5186  020472  062716  000002            3$      ADD     #2,(SP)         ;;ADJUST RETURN PC
5187  020476  000002                            RTI                     ;;RETURN
5188  020500  122716  000011            4$      CMPB    #HT,(SP)        ;;BRANCH IF <HT>
5189  020504  001430                            BEQ     8$
5190  020506  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
5191  020512  001006                            BNE     5$
5192  020514  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
5193  020516  104401                            TYPE                    ;;TYPE A CR AND LF
5194  020520  001215                            $CRLF
5195  020522  105037  020656                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
5196  020526  000755                            BR      2$              ;;GET NEXT CHARACTER
5197  020530  004737  020612            5$      JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
5198  020534  123726  001156            6$      CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS ?
5199  020540  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR
5200  020542  013746  001154                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS  NEEDED
5201                                                                    ;;AND THE NULL CHAR
5202  020546  105366  000001            7$      DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
5203  020552  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
5204  020554  004737  020612                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
5205  020560  105337  020656                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
5206  020564  000770                            BR      7$              ;;LOOP
5207
5208                                    ;HORIZONTAL TAB PROCESSOR
5209
5210  020566  112716  000040            8$      MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
5211  020572  004737  020612            9$      JSR     PC,$TYPEC       ;;TYPE A SPACE
5212  020576  132737  000007  020656            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
5213  020604  001372                            BNE     9$              ;;TAB STOP
5214  020606  005726                            TST     (SP)+           ;;POP SPACE OFF STACK
5215  020610  000724                            BR      2$              ;;GET NEXT CHARACTER
5216  020612  105777  160332            $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
5217  020616  100375                            BPL     $TYPEC
5218  020620  116677  000002  160324            MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG
5219  020626  122766  000015  000002            CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
```

```
5220  020634  001003                          BNE     1$              ;;BRANCH IF NO
5221  020636  105037  020656                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
5222  020642  000406                          BR      $TYPEX          ;;EXIT
5223  020644  122766  000012  000002  1$      CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
5224  020652  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
5225  020654  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
5226  020656  000000                  $CHARCNT  WORD  0               ;;CHARACTER COUNT STORAGE
5227  020660  000207                  $TYPEX  RTS     PC
5228
5229
5230                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
5231
5232                                  ;;**************************************************************
5233                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
5234                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT
5235                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
5236                                  ;*CALL.
5237                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5238                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
5239                                  ;*       BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
5240                                  ;*      .BYTE   M               ;;M=1 OR 0
5241                                  ;*                                      ;;1=TYPE LEADING ZEROS
5242                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
5243                                  ;*
5244                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
5245                                  ;*$TYPOS OR $TYPOC
5246                                  ;*CALL.
5247                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5248                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
5249                                  ;*
5250                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5251                                  ;*CALL
5252                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5253                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
5254
5255  020662  017646  000000          $TYPOS  MOV     @(SP),-(SP)     ;;PICKUP THE MODE
5256  020666  116637  000001  021105          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
5257  020674  112637  021107                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
5258  020700  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
5259  020704  000406                          BR      $TYPON
5260  020706  112737  000001  021105  $TYPOC  MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
5261  020714  112737  000006  021107          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
5262  020722  112737  000005  021104  $TYPON  MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
5263  020730  010346                          MOV     R3,-(SP)        ;;SAVE R3
5264  020732  010446                          MOV     R4,-(SP)        ;;SAVE R4
5265  020734  010546                          MOV     R5,-(SP)        ;;SAVE R5
5266  020736  113704  021107                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
5267  020742  005404                          NEG     R4
5268  020744  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX  ALLOWED
5269  020750  110437  021106                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
5270  020754  113704  021105                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
5271  020760  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
5272  020764  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
5273  020766  006105                  1$      ROL     R5              ;;ROTATE MSB INTO "C"
5274  020770  000404                          BR      3$              ;;GO DO MSB
5275  020772  006105                  2$      ROL     R5              ;;FORM THIS DIGIT
```

```
 5276  020774  006105                        ROL     R5
 5277  020776  006105                        ROL     R5
 5278  021000  010503                        MOV     R5,R3
 5279  021002  006103                3$      ROL     R3              ;;GET LSB OF THIS DIG'T
 5280  021004  105337  021106                DECB    $OMODE          ;;TYPE THIS DIGIT?
 5281  021010  100016                        BPL     7$              ;;BR IF NO
 5282  021012  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
 5283  021016  001002                        BNE     4$              ;;TEST FOR 0
 5284  021020  005704                        TST     R4              ;;SUPPRESS THIS 0?
 5285  021022  001403                        BEQ     5$              ;;BR IF YES
 5286  021024  005204                4$      INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 5287  021026  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 5288  021032  052703  000040          5$    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 5289  021036  110337  021102                MOVB    R3,8$           ;;SAVE FOR TYPING
 5290  021042  104401  021102                TYPE    ,8$             ;;GO TYPE THIS DIGIT
 5291  021046  105337  021104          7$    DECB    $OCNT           ;;COUNT BY 1
 5292  021052  003347                        BGT     2$              ;;BR IF MORE TO DO
 5293  021054  002402                        BLT     6$              ;;BR IF DONE
 5294  021056  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 5295  021060  000744                        BR      2$              ;;GO DO THE LAST DIGIT
 5296  021062  012605                6$      MOV     (SP)+,R5        ;;RESTORE R5
 5297  021064  012604                        MOV     (SP)+,R4        ;;RESTORE R4
 5298  021066  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 5299  021070  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 5300  021076  012616                        MOV     (SP)+,(SP)
 5301  021100  000002                        RTI                     ;;RETURN
 5302  021102     000                8$      .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 5303  021103     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 5304  021104     000                $OCNT   .BYTE   0               ;;OCTAL DIGIT COUNTER
 5305  021105     000                $OFILL  .BYTE   0               ;;ZERO FILL SWITCH
 5306  021106  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 5307
 5308                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 5309
 5310                                 ;;**************************************************************
 5311                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 5312                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT  DEPENDING ON WHETHER THE
 5313                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 5314                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER  LEADING ZEROS WILL ALWAYS BE
 5315                                 ;*REPLACED WITH SPACES.
 5316                                 ;*CALL:
 5317                                 ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 5318                                 ;*      TYPDS                   ;;GO TO THE ROUTINE
 5319
 5320  021110                         $TYPDS:
 5321  021110  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 5322  021112  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 5323  021114  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 5324  021116  010346                        MOV     R3,-(SP)        ;;PUSH P3 ON STACK
 5325  021120  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 5326  021122  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 5327  021126  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 5328  021132  100004                        BPL     1$              ;;BR IF INPUT IS POS
 5329  021134  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS
 5330  021136  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG
 5331  021144  005000                1$      CLR     R0              ;;ZERO THE CONSTANTS INDEX
```

```
5332  021146  012703  021324              MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
5333  021152  112723  000040              MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
5334  021156  005002                2$    CLR     R2              ;;CLEAR THE BCD NUMBER
5335  021160  016001  021314              MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
5336  021164  160105                3$    SUB     R1,R5           ;;FORM THIS BCD DIGIT
5337  021166  002402                      BLT     4$              ;;BR IF DONE
5338  021170  005202                      INC     R2              ;;INCREASE THE BCD DIGIT BY 1
5339  021172  000774                      BR      3$
5340  021174  060105                4$    ADD     R1,R5           ;;ADD BACK THE CONSTANT
5341  021176  005702                      TST     R2              ;;CHECK IF BCD DIGIT=0
5342  021200  001002                      BNE     5$              ;;FALL THROUGH IF 0
5343  021202  105716                      TSTB    (SP)            ;;STILL DOING LEADING 0'S?
5344  021204  100407                      BMI     7$              ;;BR IF YES
5345  021206  106316                5$    ASLB    (SP)            ;;MSD?
5346  021210  103003                      BCC     6$              ;;BR IF NO
5347  021212  116663  000001  177777      MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
5348  021220  052702  000060        6$    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
5349  021224  052702  000040        7$    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
5350  021230  110223                      MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
5351  021232  005720                      TST     (R0)+           ;;JUST INCREMENTING
5352  021234  020027  000010              CMP     R0,#10          ;;CHECK THE TABLE INDEX
5353  021240  002746                      BLT     2$              ;;GO DO THE NEXT DIGIT
5354  021242  003002                      BGT     8$              ;;GO TO EXIT
5355  021244  010502                      MOV     R5,R2           ;;GET THE LSD
5356  021246  000764                      BR      6$              ;;GO CHANGE TO ASCII
5357  021250  105726                8$    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
5358  021252  100003                      BPL     9$              ;;BR IF NO
5359  021254  116663  177777  177776      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
5360  021262  105013                9$    CLRB    (R3)            ;;SET THE TERMINATOR
5361  021264  012605                      MOV     (SP)+,R5        ;;POP STACK INTO R5
5362  021266  012603                      MOV     (SP)+,R3        ;;POP STACK INTO R3
5363  021270  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
5364  021272  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
5365  021274  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
5366  021276  104401  021324              TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
5367  021302  016666  000002  000004      MOV     2(SP),4(SP)     ;;ADJUST THE STACK
5368  021310  012616                      MOV     (SP)+,(SP)
5369  021312  000002                      RTI                     ;;RETURN TO USER
5370  021314  023420                $DTBL:  10000
5371  021316  001750                        1000
5372  021320  000144                        100.
5373  021322  000012                        10.
5374  021324  000004                $DBLK:  .BLKW   4
5375
5376                                  .SBTTL  TTY INPUT ROUTINE
5377
5378                                  ;;**********************************************************************
5379                                  .ENABL  LSB
5380  021334  000000                $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
5381  021336  000000                $TKQIN: .WORD   0               ;;INPUT POINTER
5382  021340  000000                $TKQOUT:.WORD   0               ;;OUTPUT POINTER
5383  021342  000002                $TKQSRT:.BLKB   2               ;;TTY KEYBOARD QUEUE
5384          021344                $TKQEND=.
5385
5386                                  ;*TK INITIALIZE ROUTINE
5387                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
```

```
 5388                                   ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 5389                                   ;
 5390                                   ;*CALL·
 5391                                   ;*      JSR     PC,$TKINT
 5392                                   ;*      RETURN
 5393                                   ;
 5394  021344  005037  021334           $TKINT. CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 5395  021350  012737  021342  021336           MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 5396  021356  013737  021336  021340           MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS
 5397  021364  012737  021414  000060           MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 5398  021372  012737  000200  000062           MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 5399  021400  005777  157542                   TST     @$TKB           ;;CLEAR DONE FLAG
 5400  021404  012777  000100  157532           MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 5401  021412  000207                           RTS     PC              ;;RETURN TO CALLER
 5402
 5403                                   ;*TK SERVICE ROUTINE
 5404                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 5405                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 5406                                   ;*IT IN THE QUEUE
 5407                                   ;*IF THE CHARACTER IS A "CONTROL-C" ( C) $TKINT IS CALLED AND
 5408                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
 5409                                   ;
 5410  021414  117746  157526           $TKSRV  MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 5411  021420  042716  177600                   BIC     #^C177,(SP)     ;;STRIP THE JUNK
 5412  021424  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
 5413  021430  001007                           BNE     1$              ;;BRANCH IF NO
 5414  021432  104401  022544                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C ( C)
 5415  021436  004737  021344                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 5416  021442  005726                           TST     (SP)+           ;;CLEAN UP STACK
 5417  021444  000137  004660                   JMP     START2          ;;CONTROL C RESTART
 5418  021450  021627  000007           1$      CMP     (SP),#7         ;;IS IT A CONTROL G?
 5419  021454  001004                           BNE     2$              ;;BRANCH IF NO
 5420  021456  022737  000176  001140           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 5421  021464  001500                           BEQ     6$              ;;GO TO SWR CHANGE
 5422
 5423  021466                           2$
 5424  021466  022737  000002  021334           CMP     #2,$TKCNT       ;;IS THE QUEUE FULL?
 5425  021474  001004                           BNE     3$              ;;BRANCH IF NO
 5426  021476  104401  001210                   TYPE    ,$BELL          ;;RING THE TTY BELL
 5427  021502  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 5428  021504  000451                           BR      5$              ;;EXIT
 5429  021506  021627  000023           3$      CMP     (SP),#23        ;;IS IT A CONTROL-S?
 5430  021512  001021                           BNE     32$             ;;BRANCH IF NO
 5431  021514  005077  157424                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 5432  021520  005726                           TST     (SP)+           ;;CLEAN CHAR OFF STACK
 5433  021522  105777  157416           31$     TSTB    @$TKS           ;;WAIT FOR A CHAR
 5434  021526  100375                           BPL     31$             ;;LOOP UNTIL ITS THERE
 5435  021530  117746  157412                   MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 5436  021534  042716  177600                   BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 5437  021540  022627  000021                   CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 5438  021544  001366                           BNE     31$             ;;BRANCH IF NO
 5439  021546  012777  000100  157370           MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 5440  021554  000002                           RTI                     ;;RETURN
 5441  021556  005237  021334           32$     INC     $TKCNT          ;;COUNT THIS CHARACTER
 5442  021562  021627  000140                   CMP     (SP),#140       ;;IS IT UPPER CASE?
 5443  021566  002405                           BLT     4$              ;;BRANCH IF YES
```

```
5444  021570  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
5445  021574  003002                          BGT     4$              ;;BRANCH IF YES
5446  021576  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
5447  021602  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
5448  021606  005237  021336                  INC     $TKQIN          ;;UPDATE THE POINTER
5449  021612  023727  021336  021344          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
5450  021620  001003                          BNE     5$              ;;BRANCH IF NO
5451  021622  012737  021342  021336          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
5452  021630  000002                  5$:     RTI                     ;;RETURN
5453
5454                                  ;;*****************************************************************
5455                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
5456                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
5457                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
5458                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
5459  021632  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
5460  021640  001124                          BNE     15$             ;;EXIT IF NOT
5461  021642  105777  157276                  TSTB    @$TKS           ;;IS A CHAR WAITING?
5462  021646  100121                          BPL     15$             ;;IF NOT, EXIT
5463  021650  117746  157272                  MOVB    @$TKB,-(SP)     ;;YES
5464  021654  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
5465  021660  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
5466  021664  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
5467                                                                  ;;AND EXIT
5468
5469                                  ;;*****************************************************************
5470                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
5471                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
5472                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
5473  021666  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
5474  021674  001674                          BEQ     2$              ;;BRANCH IF YES
5475  021676  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
5476  021700  004737  021344                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
5477  021704  005077  157234                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
5478  021710  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
5479
5480  021716  104401  022556                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G ( G)
5481  021722  104401  022563          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
5482  021726  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
5483  021732  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5484  021734  104401  022574                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
5485  021740  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
5486  021742  005046                          CLR     -(SP)           ;;THE NEW SWR
5487  021744  105777  157174          7$:     TSTB    @$TKS           ;;CHAR THERE?
5488  021750  100375                          BPL     7$              ;;IF NOT TRY AGAIN
5489
5490  021752  117746  157170                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
5491  021756  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
5492
5493  021762  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
5494  021766  001015                          BNE     9$              ;;BRANCH IF NOT
5495  021770  104401  022544                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C ( C)
5496  021774  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
5497  022000  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
5498  022006  001003                          BNE     8$              ;;BRANCH IF NO
5499  022010  012777  000100  157126          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
```

```
 5500  022016  000137  004660          8$      JMP     START2          ,;CONTROL-C RESTART
 5501
 5502
 5503  022022  021627  000025          9$      CMP     (SP),#25        ;;IS IT A CONTROL-U?
 5504  022026  001005                          BNE     10$             ,,BRANCH IF NOT
 5505  022030  104401  022551                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U ( U)
 5506  022034  062706  000006          20$     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 5507  022040  000737                          BR      19$             ;;LET'S TRY IT AGAIN
 5508
 5509
 5510  022042  021627  000015          10$     CMP     (SP),#15        ;;IS IT A <CR>?
 5511  022046  001022                          BNE     16$             ;;BRANCH IF NO
 5512  022050  005766  000004                  TST     4(SP)           ;,YES, IS IT THE FIRST CHAR?
 5513  022054  001403                          BEQ     11$             ,,BRANCH IF YES
 5514  022056  016677  000002  157054          MOV     2(SP),@SWR      ,,SAVE NEW SWR
 5515  022064  062706  000006          11$     ADD     #6,SP           ;;CLEAR UP STACK
 5516  022070  104401  001215          14$     TYPE    ,$CRLF          ,,ECHO <CR> AND <LF>
 5517  022074  123727  001135  000001          CMPB    $INTAG,#1       ,,RE-ENABLE TTY KBD INTERRUPTS?
 5518  022102  001003                          BNE     15$             ,,BRANCH IF NOT
 5519  022104  012777  000100  157032          MOV     #100,@$TKS      ,,RE-ENABLE TTY KBD INTERRUPTS
 5520  022112  000002          15$     RTI                             ,;RETURN
 5521  022114  004737  020612          16$     JSR     PC,$TYPEC       ,,ECHO CHAR
 5522  022120  021627  000060                  CMP     (SP),#60        ,,CHAR < 0?
 5523  022124  002420                          BLT     18$             ,,BRANCH IF YES
 5524  022126  021627  000067                  CMP     (SP),#67        ,,CHAR > 7?
 5525  022132  003015                          BGT     18$             ,,BRANCH IF YES
 5526  022134  042726  000060                  BIC     #60,(SP)+       ,,STRIP-OFF ASCII
 5527  022140  005766  000002                  TST     2(SP)           ,,IS THIS THE FIRST CHAR
 5528  022144  001403                          BEQ     17$             ,,BRANCH IF YES
 5529  022146  006316                          ASL     (SP)            ,,NO, SHIFT PRESENT
 5530  022150  006316                          ASL     (SP)            ,,    CHAR OVER TO MAKE
 5531  022152  006316                          ASL     (SP)            ,,    ROOM FOR NEW ONE
 5532  022154  005266  000002          17$     INC     2(SP)           ,,KEEP COUNT OF CHAR
 5533  022160  056616  177776                  BIS     -2(SP),(SP)     ,,SET IN NEW CHAR
 5534  022164  000667                          BR      7$              ,,GET THE NEXT ONE
 5535  022166  104401  001214          18$     TYPE    ,$QUES          ,,TYPE ?<CR><LF>
 5536  022172  000720                          BR      20$             ,,SIMULATE CONTROL-U
 5537                          DSABL   LSB
 5538
 5539
 5540                          ,,*****************************************************************
 5541                          ,*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 5542                          ,*CALL
 5543                          ,*      RDCHR                   ,,GET A CHARACTER FROM THE QUEUE
 5544                          .*      RETURN HERE             ,,CHARACTER IS ON THE STACK
 5545                          ,*                              ,,WITH PARITY BIT STRIPPED OFF
 5546                          ,
 5547
 5548  022174  011646          $PDCHR  MOV     (SP),-(SP)      ,,PUSH DOWN THE PC AND
 5549  022176  016666  000004  000002          MOV     4(SP),2(SP)     ,,THE PS
 5550  022204  005066  000004                  CLR     4(SP)           ,,GET READY FOR A CHARACTER
 5551  022210  005046                          CLR     -(SP)           ,,PUT NEW PS ON STACK
 5552  022212  012746  022220                  MOV     #64$,-(SP)      ,,PUT NEW PC ON  STACK
 5553  022216  000002                          RTI                     ,,POP NEW PC AND PS
 5554  022220                          64$
 5555  022220  005737  021334          1$      TST     $TKCNT          ,,WAIT ON A CHARACTER
```

```
5556  022224  001775                          BEQ     1$
5557  022226  005337  021334                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
5558  022232  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
5559  022240  005237  021340                  INC     $TKQOUT         ;;UPDATE THE POINTER
5560  022244  023727  021340  021344          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
5561  022252  001003                          BNE     2$              ;;BRANCH IF NO
5562  022254  012737  021342  021340          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
5563  022262  000002                  2$      RTI                     ;;RETURN
5564                                  ;;**********************************************************************
5565                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
5566                                  ;*CALL:
5567                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
5568                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
5569                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
5570
5571  022264  010346                  $RDLIN  MOV     R3,-(SP)        ;;SAVE R3
5572  022266  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
5573  022270  012703  022520          1$      MOV     #$TTYIN,R3      ;;GET ADDRESS
5574  022274  022703  022544          2$      CMP     #$TTYIN+20,R3   ;;BUFFER FULL?
5575  022300  101456                          BLOS    4$              ;;BR IF YES
5576  022302  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
5577  022304  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
5578  022306  122713  000177          10$     CMPB    #177,(R3)       ;;IS IT A RUBOUT
5579  022312  001022                          BNE     5$              ;;BR IF NO
5580  022314  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
5581  022316  001007                          BNE     6$              ;;BR IF NO
5582  022320  112737  000134  022516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
5583  022326  104401  022516                  TYPE    ,9$
5584  022332  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
5585  022336  005303                  6$      DEC     R3              ;;BACKUP BY ONE
5586  022340  020327  022520                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
5587  022344  103434                          BLO     4$              ;;BR IF YES
5588  022346  111337  022516                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR
5589  022352  104401  022516                  TYPE    ,9$             ;;GO TYPE
5590  022356  000746                          BR      2$              ;;GO READ ANOTHER CHAR
5591  022360  005716                  5$      TST     (SP)            ;;RUBOUT KEY SET?
5592  022362  001406                          BEQ     7$              ;;BR IF NO
5593  022364  112737  000134  022516          MOVB    #'\,9$          ;;TYPE A BACK SLASH
5594  022372  104401  022516                  TYPE    ,9$
5595  022376  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
5596  022400  122713  000025          7$      CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
5597  022404  001003                          BNE     8$              ;;BR IF NO
5598  022406  104401  022551                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
5599  022412  000726                          BR      1$              ;;GO START OVER
5600  022414  122713  000022          8$      CMPB    #22,(R3)        ;;IS CHARACTER A " R"?
5601  022420  001011                          BNE     3$              ;;BRANCH IF NO
5602  022422  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
5603  022424  104401  001215                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
5604  022430  104401  022520                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
5605  022434  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
5606  022436  104401  001214          4$      TYPE    ,$QUES          ;;TYPE A '?'
5607  022442  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
5608  022444  111337  022516          3$      MOVB    (R3),9$         ;;ECHO THE CHARACTER
5609  022450  104401  022516                  TYPE    ,9$
5610  022454  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
5611  022460  001305                          BNE     2$              ;;LOOP IF NOT RETURN
```

```
5612  022462  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
5613  022466  104401  001216                  TYPE    ,$LF            ;;TYPE A LINE FEED
5614  022472  005726                          TST     (SP)+           ;,CLEAN RUBOUT KEY FROM THE STACK
5615  022474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
5616  022476  011646                          MOV     (SP),-(SP)      ,;ADJUST THE STACK AND PUT ADDRESS OF THE
5617  022500  016666  000004  000002          MOV     4(SP),2(SP)     ;,      FIRST ASCII CHARACTER ON IT
5618  022506  012766  022520  000004          MOV     #$TTYIN,4(SP)
5619  022514  000002                          RTI                     ,;RETURN
5620  022516     000                  9$      .BYTE   0               ;;STORAGE FOR ASCII CHAR  TO TYPE
5621  022517     000                          .BYTE   0               ;;TERMINATOR
5622  022520  000024                  $TTYIN: .BLKB   20.             ;;RESERVE 20. BYTES FOR TTY INPUT
5623  022544  041536  005015     000  $CNTLC: .ASCIZ  / C/<15><12>    ;;CONTROL "C"
5624  022551     136  006525  000012  $CNTLU: .ASCIZ  / U/<15><12>    ;;CONTROL "U"
5625  022556  043536  005015     000  $CNTLG: .ASCIZ  / G/<15><12>    ;;CONTROL "G"
5626  022563     015  051412  051127  $MSWR.  .ASCIZ  <15><12>/SWR = /
5627  022570  036440  000040
5628  022574  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
5629  022602  020075     000
5630          022606                   EVEN
5631
5632                                   SBTTL  SCOPE HANDLER ROUTINE
5633
5634                                  ;,**************************************************************
5635                                  ,*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS.  IT WILL INCREMENT
5636                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG (DISPLAY<7 0>)
5637                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15.08>
5638                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE
5639                                  ;*SW14=1         LOOP ON TEST
5640                                  ;*SW11=1         INHIBIT ITERATIONS
5641                                  ,*SW09=1         LOOP ON ERROR
5642                                  ,*CALL
5643                                  ,*       SCOPE           ,;SCOPE=IOT
5644
5645  022606                  $SCOPE:
5646  022606  104407                          CKSWR                   ;,TEST  FOR CHANGE IN SOFT-SWR
5647  022610  032777  040000  156322  1$:     BIT     #BIT14,@SWR     ,,LOOP ON PRESENT TEST?
5648  022616  001101                          BNE     $OVER           ;,YES IF SW14=1
5649                                  ,#####START OF CODE FOR THE XOR TESTER#####
5650  022620  000416                  $XTSTR  BR      6$              ,;IF RUNNING ON THE "XOR" TESTER CHANGE
5651                                                                  ;;THIS INSTRUCTION TO A 'NOP" (NOP=240)
5652  022622  013746  000004                  MOV     @#ERRVEC,-(SP)  ;,SAVE THE CONTENTS OF THE ERROR VECTOR
5653  022626  012737  022646  000004          MOV     #5$,@#ERRVEC    ,;SET FOR TIMEOUT
5654  022634  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
5655  022640  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
5656  022644  000453                          BR      $SVLAD          ;;GO TO THE NEXT TEST
5657  022646  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
5658  022650  012637  000004                  MOV     (SP)+,@#ERRVEC  ;,RESTORE THE ERROR VECTOR
5659  022654  000413                          BR      7$              ,;LOOP ON THE PRESENT TEST
5660  022656                  6$ ,#####END OF CODE FOR THE XOR TESTER#####
5661  022656  105737  001103          2$      TSTB    $ERFLG          ;,HAS AN ERROR OCCURRED?
5662  022662  001421                          BEQ     3$              ;,BR IF NO
5663  022664  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ,,MAX. ERRORS FOR THIS TEST OCCURRED?
5664  022672  101015                          BHI     3$              ,,BR IF NO
5665  022674  032777  001000  156236          BIT     #BIT09,@SWR     ,,LOOP ON ERROR?
5666  022702  001404                          BEQ     4$              ,,BR IF NO
5667  022704  013737  001110  001106  7$      MOV     $LPERR,$LPADR   ,,SET LOOP ADDRESS TO LAST SCOPE
```

```
5668  022712  000443                          BR      $OVER
5669  022714  105037  001103          4$      CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
5670  022720  005037  001204                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
5671  022724  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
5672  022726  032777  004000  156204  3$      BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
5673  022734  001011                          BNE     1$              ;;BR IF YES
5674  022736  005737  001100                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
5675  022742  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
5676  022744  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
5677  022750  023737  001204  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
5678  022756  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
5679  022760  012737  000001  001104  1$      MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
5680  022766  013737  023036  001204          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
5681  022774  105237  001102          $SVLAD  INCB    $TSTNM          ;;COUNT TEST NUMBERS
5682  023000  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
5683  023004  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
5684  023010  005037  001206                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
5685  023014  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
5686  023022  013777  001102  156112  $OVER   MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
5687  023030  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
5688  023034  000002                          RTI                     ;;FIXES PS
5689  023036  000001                  $MXCNT  1                       ;;MAX. NUMBER OF ITERATIONS
5690
5691                                   SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
5692
5693                                  ;;*******************************************************************
5694                                  ;*SAVE R0-R5
5695                                  ;*CALL
5696                                  ;*      SAVREG
5697                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE
5698                                  ;*
5699                                  ;*TOP---(+16)
5700                                  ;* +2---(+18)
5701                                  ;* +4---R5
5702                                  ;* +6---R4
5703                                  ;* +8---R3
5704                                  ;*+10---R2
5705                                  ;*+12---R1
5706                                  ;*+14---R0
5707
5708  023040                          $SAVREG:
5709  023040  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
5710  023042  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
5711  023044  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
5712  023046  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5713  023050  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
5714  023052  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
5715  023054  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
5716  023060  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
5717  023064  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
5718  023070  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
5719  023074  000002                          RTI
5720
5721                                  ;*RESTORE R0-R5
5722                                  ;*CALL
5723                                  ;*      RESREG
```

```
5724  023076                          $RESREG:
5725  023076  012666  000022                  MOV     (SP)+,22(SP     ;;RESTORE PC OF CALL
5726  023102  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
5727  023106  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
5728  023112  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
5729  023116  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
5730  023120  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
5731  023122  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
5732  023124  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
5733  023126  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
5734  023130  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
5735  023132  000002                          RTI
5736
5737                                   .SBTTL  TRAP DECODER
5738
5739                                  ;;*****************************************************************
5740                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
5741                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
5742                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
5743                                  ;*GO TO THAT ROUTINE.
5744
5745  023134  010046                  $TRAP   MOV     R0,-(SP)        ;;SAVE R0
5746  023136  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
5747  023142  005740                          TST     -(R0)           ;;BACKUP BY 2
5748  023144  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
5749  023146  006300                          ASL     R0              ;;POSITION FOR INDEXING
5750  023150  016000  023170                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
5751  023154  000200                          RTS     R0              ;;GO TO ROUTINE
5752
5753
5754                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
5755
5756  023156  011646                  $TRAP2  MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
5757  023160  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
5758  023166  000002                          RTI                     ;;RESTORE THE PSW
5759
5760                                   .SBTTL  TRAP TABLE
5761
5762                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
5763                                  ;*BY THE "TRAP" INSTRUCTION
5764
5765                                  ;       ROUTINE
5766                                  ;       -------
5767  023170  023156                  $TRPAD: .WORD   $TRAP2
5768  023172  020442                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
5769  023174  020706                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
5770  023176  020662                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
5771  023200  020722                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
5772  023202  021110                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
5773
5774  023204  021722                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
5775
5776  023206  021632                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
5777  023210  022174                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
5778  023212  022264                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
5779  023214  023040                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
```

```
 5780  023216  023076                          $RESREG ,,CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
 5781
 5782                                   SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
 5783
 5784                                  ;,*****************************************************************
 5785                                  ,*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINAPY NUMBER TO AN
 5786                                  ,*UNSIGNED DECIMAL ASCIZ NUMBER.
 5787                                  ,*CALL
 5788                                  ,*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
 5789                                  ;*      JSR     PC,@#$SB2D      ,;CALL
 5790                                  ,*      RETURN                  ,;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
 5791
 5792
 5793  023220  016637  000002  023250  $SB2D   MOV     2(SP),1$        ;,SAVE BINARY NUMBER
 5794  023226  012746  023250                  MOV     #1$,-(SP)       ;;SET POINTER
 5795  023232  004737  023254                  JSR     PC,@#$DB2D      ;,CALL DOUBLE LENGTH CONVERT
 5796  023236  062716  000005                  ADD     #5,(SP)         ,;ONLY ALLOW FIVE CHARACTERS
 5797  023242  012666  000002                  MOV     (SP)+,2(SP)     ,;PICKUP POINTER
 5798  023246  000207                          RTS     PC              ,;RETURN
 5799  023250  000000  000000          1$       WORD   0,0
 5800
 5801                                   SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
 5802
 5803                                  ,,*****************************************************************
 5804                                  ,*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
 5805                                  ,*DECIMAL (ASCII) NUMBER.  THE SIGN OF THE BINARY NUMBER MUST BE
 5806                                  ,*POSITIVE.
 5807                                  ,*CALL
 5808                                  ,*      MOV     #PNTR,-(SP)     ,,POINTER TO LOW WORD OF BINARY NUMBER
 5809                                  ,*      JSR     PC,@#$DB2D
 5810                                  ,*      RETURN                  ,,THE FIRST ADDRESS OF ASCIZ
 5811                                                                  ,,IS ON THE STACK
 5812
 5813
 5814  023254  104412                  $DB2D   SAVREG                  ,,SAVE REGISTERS
 5815  023256  016602  000002                  MOV     2(SP),R2        ,,PICKUP THE DATA POINTER
 5816  023262  012700  023434                  MOV     #$DECVL,R0      ,,GET ADDRESS OF "$DECVL" STRING
 5817  023266  010066  000002                  MOV     R0,2(SP)        ,,PUT ADDRESS OF ASCIZ STRING ON STACK
 5818  023272  012201                          MOV     (R2)+,R1        ,,PICKUP THE BINARY NUMBER
 5819  023274  012202                          MOV     (R2)+,R2
 5820  023276  012737  000012  023352          MOV     #10.,4$         ,,SET UP TO DO 10 CONVERSIONS
 5821  023304  012704  023364                  MOV     #$TNPWR,R4      ,,ADDRESS OF TEN POWER
 5822  023310  012705  023366                  MOV     #$TNPWR+2,R5
 5823  023314  005003                  1$      CLR     R3              ,,CLEAR PARTIAL
 5824  023316  161401                  2$      SUB     (R4),R1         ,,SUBTRACT TEN POWER
 5825  023320  005602                          SBC     R2
 5826  023322  161502                          SUB     (R5),R2
 5827  023324  002402                          BLT     3$              ,;BR IF TEN POWER TO LARGE
 5828  023326  005203                          INC     R3              ,,ADD 1 TO PARTIAL
 5829  023330  000772                          BR      2$              ,,LOOP
 5830  023332  062401                  3$      ADD     (R4)+,R1        ;,RESTORE SUBTRACTED VALUE
 5831  023334  005502                          ADC     R2
 5832  023336  062402                          ADD     (R4)+,R2
 5833  023340  022525                          CMP     (R5)+,(R5)+     ,,MOVE TO NEXT TEN POWER
 5834  023342  052703  000060                  BIS     #'0,R3          ,,CHANGE PARTIAL TO ASCII
 5835  023346  110320                          MOVB    R3,(R0)+        ,,SAVE IT
```

```
5836  023350  005327                    DEC     (PC)+           ;;DONE?
5837  023352  000000              4$     WORD    0
5838  023354  001357                    BNE     1$              ;;BR IF NO
5839  023356  105020                    CLRB    (R0)+           ;;TERMINATOR
5840  023360  104413                    RESREG                  ;;RESTORE REGISTERS
5841  023362  000207                    RTS     PC              ;;RETURN
5842  023364  145000              $TNPWR: 145000                  ;;1 OE09
5843  023366  035632                    35632
5844  023370  160400                    160400                  ;;1 OE08
5845  023372  002765                    2765
5846  023374  113200                    113200                  ;;1 OE07
5847  023376  000230                    230
5848  023400  041100                    041100                  ;;1 OE06
5849  023402  000017                    17
5850  023404  103240                    103240                  ;;1 OE05
5851  023406  000001                    1
5852  023410  023420                    23420                   ;;1 OE04
5853  023412  000000                    0
5854  023414  001750                    1750                    ;;1 OE03
5855  023416  000000                    0
5856  023420  000144                    144                     ;;1 OE02
5857  023422  000000                    0
5858  023424  000012                    12                      ;;1 OE01
5859  023426  000000                    0
5860  023430  000001                    1                       ;;1 OE00
5861  023432  000000                    0
5862  023434  000014              $DECVL  BLKB    12              ;;RESERVE STORAGE FOR ASCIZ STRING
5863
5864                                    SBTTL   TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
5865
5866                                    ;;**************************************************************
5867                                    ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
5868                                    ;*LEADING NUMBERS
5869                                    ;*CALL
5870                                    ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
5871                                    ;*      JSR     PC,@#$SUPRS
5872
5873
5874  023450  010046              $SUPRS  MOV     R0,-(SP)        ;;SAVE R0
5875  023452  016600  000004              MOV     4(SP),R0        ;;PICKUP THE POINTER
5876  023456  105710              1$      TSTB    (R0)            ;;TERMINATEOR?
5877  023460  001403                      BEQ     2$              ;;BR IF YES
5878  023462  122720  000060              CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
5879  023466  001773                      BEQ     1$              ;;BR IF YES
5880  023470  005300              2$      DEC     R0              ;;BACKUP BY "1"
5881  023472  010037  023500              MOV     R0,3$           ;;SAVE FOR TYPING
5882  023476  104401                      TYPE                    ;;GO TYPE
5883  023500  000000              3$      WORD    0               ;;ASCIZ POINTER GOES HERE
5884  023502  012600                      MOV     (SP)+,R0        ;;RESTOPE R0
5885  023504  012616                      MOV     (SP)+,(SP)      ;;RESTORE THE STACK
5886  023506  000207                      RTS     PC              ;;RETURN
5887
5888                                    SBTTL   RANDOM NUMBER GENERATOR ROUTINE
5889
5890                                    ;;**************************************************************
5891                                    ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
```

```
5892                                    ,*WITH A RANGE OF 0 TO 2(+33)-1
5893                                    ,*CALL
5894                                    ,*      JSR     PC,$RAND        ,,CALL THE ROUTINE
5895                                    ,*      RETURN                  ;,RETURN HERE THE RANDOM
5896                                    ,*                              ;,NUMBER WILL BE IN
5897                                    ,*                              ,,$HINUM,$LONUM
5898
5899  023510                            $RAND
5900  023510  010046                            MOV     R0,-(SP)        ,,PUSH R0 ON STACK
5901  023512  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
5902  023514  010246                            MOV     R2,-(SP)        ,;PUSH R2 ON STACK
5903  023516  013700  023610                    MOV     $LONUM,R0       ,,SET R0 WITH LOW
5904  023522  013701  023606                    MOV     $HINUM,R1       ,,SET R1 WITH HIGH
5905  023526  012702  177771                    MOV     #-7,R2          ,,SET SHIFT COUNT
5906  023532  006300                    1$      ASL     R0              ,,SHIFT R0 LEFT AND
5907  023534  006101                            ROL     R1              ,,ROTATE CARRY INTO R1 AND
5908  023536  005202                            INC     R2              ,;CHECK FOR DONE
5909  023540  001374                            BNE     1$              ,,CONTINUE SHIFT LOOP
5910  023542  063700  023610                    ADD     $LONUM,R0       ,,ADD NUMBER TO MAKE X 129
5911  023546  005501                            ADC     R1              ,;PROPOGATE CARRY
5912  023550  063701  023606                    ADD     $HINUM,R1       ,,ADD NUMBER TO MAKE X 129
5913  023554  062700  001057                    ADD     #1057,R0        ,,ADD LOW CONSTANT
5914  023560  005501                            ADC     R1              ,,PROPOGATE CARRY
5915  023562  062701  047401                    ADD     #47401,R1       ,,ADD HIGH CONSTANT
5916  023566  010037  023610                    MOV     R0,$LONUM       ,,SAVE R0
5917  023572  010137  023606                    MOV     R1,$HINUM       ,,SAVE R1
5918  023576  012602                            MOV     (SP)+,R2        ,,POP STACK INTO R2
5919  023600  012601                            MOV     (SP)+,R1        ,,POP STACK INTO R1
5920  023602  012600                            MOV     (SP)+,R0        ,,POP STACK INTO R0
5921  023604  000207                            RTS     PC              ,,RETURN
5922  023606  176543                    $HINUM  .WORD   176543
5923  023610  123456                    $LONUM   WORD   123456
5924
5925                                     SBTTL  INTEGER DIVIDE ROUTINE
5926
5927                                    , ************************************************************************
5928                                    ,*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
5929                                    ,*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
5930                                    ,*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER
5931                                    ,*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
5932                                    ,*SAVE SIGN AS THE DIVIDEND.
5933                                    ,*CALL:
5934                                    ,*      MOV     LOW DIVIDEND,-(SP) ,,THE HIGH DIVIDEND MUST BE < 1/2
5935                                    ,*      MOV     HIGH DIVIDEND,-(SP).    AS LARGE AS THE DIVISOR
5936                                    ,*      MOV     DIVISOR,-(SP)
5937                                    ,*      JSR     PC,$DIV
5938                                    ,*      RETURN                  ,,QUOTIENT & REMAINDER ARE ON THE STACK
5939                                    ,*      "V"=0   IMPLIES NO ERROR
5940                                    ,*      "V"=1   IMPLIES ERROR OCCURRED
5941                                    ,*              "C"=0   DIVIDE OVERFLOW OCCURRED
5942                                    ,*              "C"=1   ATTEMPTED TO DIVIDE BY ZERO
5943                                    ,*
5944                                    ,*
5945                                    ,*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
5946                                    ,*      -----   --------        --------        -------------
5947                                    ,*      TOP     REMAINDER       ALL ZEROS       ALL ONES
```

```
                                        .X      +2      QUOTIENT        ALL ZEROS       ALL ONES

023612                                  $DIV
023612  104400                          TRAP                            ;;PUSH OLD PSW AND PC ON STACK
023614  042716  000017                  BIC     #17,(SP)                ;;STRIP AWAY CONDITION CODES
023620  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
023622  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
023624  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
023626  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
023630  005046                          CLR     -(SP)                   ;;SAVE A PLACE FOR SIGNS
023632  012746  000021                  MOV     #17 ,-(SP)              ;;SETUP THE ITERATION COUNTER
023636  016601  000024                  MOV     24(SP),R1               ;;PICKUP THE DIVIDEND
023642  016600  000022                  MOV     22(SP),R0
023646  100005                          BPL     1$                      ;;CHECK THE SIGN
023650  105366  000003                  DECB    3(SP)                   ;;KEEP TRACK OF THE SIGN
023654  005400                          NEG     R0                      ;;AND NEGATE THE ORIGINAL
023656  005401                          NEG     R1                      ;;NUMBER
023660  005600                          SBC     R0
023662  016602  000020          1$      MOV     20(SP),R2               ;;PICKUP THE DIVISOR
023666  002407                          BLT     2$                      ;;CHECK THE SIGN
023670  003011                          BGT     3$                      ;;DIVISOR OF 0 IS A NO-NO
023672  052766  000003  000014          BIS     #3,14(SP)               ;;SET "V" & "C"
023700  012700  177777                  MOV     #-1,R0                  ;;SET REMAINDER TO ALL ONES
023704  000424                          BR      7$                      ;;EXIT
023706  005266  000002          2$      INC     2(SP)                   ;;KEEP TRACK OF DIVISORS SIGN
023712  000401                          BR      4$
023714  005402                  3$      NEG     R2                      ;;NEGATE THE ORIGINAL NUMBER
023716  000241                  4$      CLC                             ;;CLEAR "C"
023720  000405                          BR      6$                      ;;START FORMING QUOTIENT
023722  006100                  5$      ROL     R0                      ;;POSITION MSB'S
023724  010003                          MOV     R0,R3                   ;;COPY
023726  060203                          ADD     R2,R3                   ;;COMPARE DIVIDEND & DIVISOR
023730  103001                          BCC     6$                      ;;BR IF DIVIDEND > DIVISOR
023732  010300                          MOV     R3,R0                   ;;REMAINDER AFTER THIS LOOP
023734  006101                  6$      ROL     R1                      ;;QUOTIENT BIT ENTERS HERE
023736  005316                          DEC     (SP)                    ;;DONE?
023740  001370                          BNE     5$                      ;;BR IF NO
023742  005701                          TST     R1                      ;;OVERFLOW?
023744  100005                          BPL     8$                      ;;BR IF NO
023746  052766  000002  000014          BIS     #2,14(SP)               ;;SET "V" IN RETURN STATUS WORD
023754  005000                          CLR     R0                      ;;SET REMAINDER TO ALL ZEROS
023756  010001                  7$      MOV     R0,R1                   ;;COPY REMAINDER INTO QUOTIENT
023760  005726                  8$      TST     (SP)+                   ;;CLEAR COUNTER FROM STACK
023762  005716                          TST     (SP)                    ;;REMAINDER SIGN CORRECTION NEEDED?
023764  002004                          BGE     9$                      ;;BR IF NO
023766  005400                          NEG     R0                      ;;NEGATE REMAINDER
023770  105066  000001                  CLRB    1(SP)                   ;;CLEAR SIGN
023774  005316                          DEC     (SP)                    ;;BUT DON'T FORGET QUOTIENT
023776  005726                  9$      TST     (SP)+                   ;;QUOTIENT SIGN CORRECTION NEEDED?
024000  001401                          BEQ     10$                     ;;BR IF NO
024002  005401                          NEG     R1                      ;;NEGATE QUOTIENT
024004  010166  000020          10$     MOV     R1,20(SP)               ;;RETURN QUOTIENT AND
024010  010066  000016                  MOV     R0,16(SP)               ;;REMAINDER TO USER
024014  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
024016  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
024020  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
```

```
 6004  024022  012600                        MOV     (SP)+,R0        ,,POP STACK INTO R0
 6005  024024  012666  000002                MOV     (SP)+,2(SP)     ,,SETUP TO RETURN CONDITION CODES
 6006  024030  000002                        RTI                     ,,RETURN
 6007
 6008                                  SBTTL  *** PROGRAM SUBROUTINES ***
 6009
 6010                                 ,SET "LPTAVL" TO THE PROPER STATE.
 6011                                 , LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
 6012                                 , LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
 6013                                 ,CALL
 6014                                 ,       JSR     PC,@#LP.AVL
 6015                                 ,       RETURN
 6016
 6017  024032  005037  001230        LP AVL  CLR     @#LPTAVL        ,START WITH NO PRINTER AVAIABLE
 6018  024036  012737  024062  000004        MOV     #1$,@#ERRVEC    ;SETUP THE TIMEOUT VECTOR
 6019  024044  005037  000006                CLR     @#ERRVEC+2
 6020  024050  005777  155344                TST     @LPS            ,IS THERE A LINE PRINTER?
 6021  024054  005237  001230                INC     @#LPTAVL        ,YES--SET AVAILABLE SWITCH
 6022  024060  000401                        BR      2$
 6023  024062  022626                1$      CMP     (SP)+,(SP)+     ,NO--POP STACK
 6024  024064  012737  000006  000004  2$    MOV     #ERRVEC+2,@#ERRVEC ,RESTORE TIMEOUT VECTOR
 6025  024072  000207                        RTS     PC              ,RETURN
 6026
 6027                                 ,THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
 6028                                 ,AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
 6029                                 ,"CLKSTA" WILL INDICATE THE CLOCK TYPE
 6030                                 , 0= NO CLOCK
 6031                                 ,+1= KW11-P
 6032                                 ;-1= KW11-L
 6033                                 ,THIS ROUTINE WILL ALSO SETUP "TICKMS" (TIME
 6034                                 ,PER CLOCK TICK IN MILLISECONDS) AND "TICKUS"
 6035                                 ,(TIME PER CLOCK TICK IN MICROSECONDS) AS
 6036                                 ,PER SW00.
 6037                                 ,SW00=0 -- 60HZ
 6038                                 ,SW00=1 -- 50HZ
 6039                                 ,CALL
 6040                                 ,       JSR     PC,@#ST CLK
 6041                                 ,       RETURN
 6042
 6043  024074  010146                ST CLK  MOV     R1,-(SP)        ,SAVE R1
 6044  024076  012701  000006                MOV     #ERRVEC+2,R1    ,SAVE AND SETUP TIMEOUT VECTOR
 6045  024102  011146                        MOV     (R1),-(SP)
 6046  024104  005011                        CLR     (R1)            ,LEVEL 0
 6047  024106  014146                        MOV     -(R1),-(SP)
 6048  024110  012711  024140                MOV     #1$,(R1)        ,GO TO 1$ ON TIMEOUT
 6049  024114  005037  001244                CLR     CLKSTA          ,SET CLOCK STATUS TO NO CLOCK
 6050  024120  005777  155254                TST     @PKCS           ;IS THERE A KW11-P?
 6051  024124  012737  000001  001244        MOV     #1,CLKSTA       ,YES--SET STATUS TO KW11-P
 6052  024132  004737  024242                JSR     PC,ST.PCLK      ,START THE KW11-P
 6053  024136  000414                        BR      3$              ;GO TO EXIT
 6054  024140  022626                1$      CMP     (SP)+,(SP)+     ,CLEAN UP THE STACK
 6055  024142  012711  024166                MOV     #2$,(R1)        ,IF TIMEOUT GO TO 2$
 6056  024146  005777  155240                TST     @LKS            ;IS THERE A KW11-L?
 6057  024152  012737  177777  001244        MOV     #-1,CLKSTA      ,YES-- SET STATUS TO KW11-L
 6058  024160  004737  024304                JSR     PC,ST LCLK      ,START THE KW11-L
 6059  024164  000401                        BR      3$              ,EXIT
```

```
6060  024166  022626                    2$      CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
6061  024170  012621                    3$.     MOV     (SP)+,(R1)+     ;RESTORE THE TIMEOUT VECTOR
6062  024172  012621                            MOV     (SP)+,(R1)+
6063  024174  012601                            MOV     (SP)+,R1        ;RESTORE R1
6064  024176  032737  000100  001220            BIT     #SW06,@#C SWR   ;50HZ OR 60HZ?
6065  024204  001407                            BEQ     4$              ,BRANCH IF 60
6066  024206  012737  000020  001246            MOV     #20,@#TICKMS    ;SETUP TIME PER
6067  024214  012737  047040  001250            MOV     #20000.,@#TICKUS ;TICK FOR 50HZ
6068  024222  000406                            BR      5$
6069  024224  012737  000016  001246  4$        MOV     #16,@#TICKMS    ;SETUP TIME PER
6070  024232  012737  040432  001250            MOV     #16666 ,@#TICKUS ;TICK FOR 60HZ
6071  024240  000207                    5$      RTS     PC              ,RETURN
6072
6073  024242                            ST PCLK
6074  024242  032737  000040  001220            BIT     #SW05,@#C SWR   ,ALLOW SOFTWARE TIMEOUTS?
6075  024250  001014                            BNE     1$              ,NO--BRANCH
6076  024252  012777  024340  155114            MOV     #SRVCLK,@PKV    ,SETUP THE KW11-P VECTOR
6077  024260  012777  000300  155110            MOV     #300,@PKV+2
6078  024266  012777  000001  155106            MOV     #1,@PKB         ;COUNT ONE TICK
6079  024274  012777  000115  155076            MOV     #115,@PKCS      ,"INT EN.",COUNT DOWN", "MODE 1 (REPEAT)",
6080                                                                    ,"LINE FREQ", AND "RUN"
6081  024302  000207                    1$      RTS     PC              ,RETURN
6082
6083  024304                            ST LCLK
6084  024304  032737  000040  001220            BIT     #SW05,@#C SWR   ,ALLOW SOFTWARE TIMEOUTS?
6085  024312  001011                            BNE     1$              ,NO--BRANCH
6086  024314  012777  024340  155064            MOV     #SRVCLK,@LKV    ;SETUP THE KW11-L VECTOR
6087  024322  012777  000300  155060            MOV     #300,@LKV+2
6088  024330  012777  000100  155054            MOV     #100,@LKS       ,START THE KW11-L
6089  024336  000207                    1$      RTS     PC              ,RETURN
6090
6091  024340  013746  001246            SRVCLK. MOV     @#TICKMS,-(SP)  ,TIME PER TICK IN MILLISECONDS
6092  024344  004737  040734                    JSR     PC,@#RPTMR      ;COUNT THE ELASPED TIME
6093  024350  000002                            RTI                     ,RETURN AFTER INTERRUPT
6094
6095                                    ,THIS ROUTINE SETS UP DEFAULT PARAMETER VALUES WHEN THE PROGRAM IS
6096                                    ,STARTED OR WHEN THE VALUE OF BIT00 IN 'C SWR' IS CHANGED
6097                                    ,CALL
6098                                    ,       JSR     PC,LODFLT
6099                                    ,       RETURN
6100
6101  024352                            LODFLT:
6102  024352  010046                            MOV     R0,-(SP)        ;,PUSH R0 ON STACK
6103  024354  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
6104  024356  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
6105  024360  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
6106  024362  012737  176777  001234            MOV     #176777,TSTNMS  ;SELECT TESTS 0-10, 12-17
6107  024370  012737  000001  001236            MOV     #1,TSTNMS+2     ;SET SELECT BIT FOR TEST 20
6108  024376  012700  001664                    MOV     #DFLT,R0        ;DEFAULT PARAMETERS POINTER
6109  024402  012701  002330                    MOV     #PRM0,R1        ;TABLE POINTER
6110  024406  010102                            MOV     R1,R2           ;STOP ADDRESS
6111  024410  012021                    1$      MOV     (R0)+,(R1)+     ;MOVE DEFAULT PARAMETERS INTO
6112  024412  020002                            CMP     R0,R2           ;RUN TIME TABLES ** DONE?
6113  024414  103775                            BLO     1$              ,NO--BRANCH
6114  024416  012700  003504                    MOV     #PAT8,R0        ,PAT0 DEFAULTS TO PATTERN 8
6115  024422  012701  003104                    MOV     #PAT0,R1
```

```
6116  024426  012021                    2$      MOV     (R0)+,(R1)+
6117  024430  020027  003544                    CMP     R0,#PAT9
6118  024434  103774                            BLO     2$
6119  024436  032737  000001  001220            BIT     #BIT00,C.SWR    ;16 BIT MODE ?
6120  024444  001012                            BNE     3$              ;BR IF 18
6121  024446  012737  000025  001630            MOV     #21.,PRMLMT+22  ;SET 'FS' LIMIT TO 21.
6122  024454  012737  000025  001632            MOV     #21.,PRMLMT+24  ;SET 'LS' LIMIT TO 21
6123  024462  012737  165000  001352            MOV     #-(256.*22.),TRCKWC  ;WORD COUNT FOR A 16 BIT TRACK
6124  024470  000411                            BR      4$              ;CONTINUE
6125  024472  012737  000023  001630  3$        MOV     #19.,PRMLMT+22  ;SET 'FS' LIMIT TO 19.
6126  024500  012737  000023  001632            MOV     #19.,PRMLMT+24  ;SET 'LS' LIMIT TO 19
6127  024506  012737  166000  001352            MOV     #-(256*20.),TRCKWC  ;WORD COUNT FOR COUNT FOR AN 18 BIT TRACK
6128  024514  012701  001536            4$      MOV     #PRMPT,R1       ;ADDRESS OF PARAMETER POINTER TABLE
6129  024520  005711                    5$      TST     (R1)            ;END OF THE TABLE ?
6130  024522  001425                            BEQ     8$              ,BR IF END
6131  024524  032731  002000                    BIT     #BIT10,@(R1)+   ;'LS' SELECTED ?
6132  024530  001773                            BEQ     5$              ,BR IF NOT
6133  024532  016102  177776                    MOV     -2(R1),R2       ,PARAMETER TABLE ADDRESS
6134  024536  011246                            MOV     (R2),-(SP)      ;PARAMETER ALLOCATION BITS
6135  024540  012703  000013                    MOV     #11 ,R3         ,NUMBER OF PARAMETERS (MAXIMUM) BEFORE 'LS'
6136  024544  006216                    6$      ASR     (SP)            ,COUNT THE PARAMETER
6137  024546  103002                            BCC     7$              ,BR IF NOT USED
6138  024550  062702  000002                    ADD     #2,R2           ,INCREMENT THE PARAMETER TABLE ADDRESS
6139  024554  005303                    7$      DEC     R3              ,COUNT THE PARAMETER
6140  024556  001372                            BNE     6$              ;BR IF NOT THERE YET
6141  024560  005726                            TST     (SP)+           ,CORRECT THE STACK POINTER
6142  024562  021237  001630                    CMP     (R2),PRMLMT+22  ;IS 'LS' TOO LARGE FOR THE MODE SELECTED ?
6143  024566  101754                            BLOS    5$              ,BR IF NOT
6144  024570  013712  001630                    MOV     PRMLMT+22,(R2)  ,RESET VALUE FOR MODE USED
6145  024574  000751                            BR      5$              ,CONTINUE
6146  024576                            8$
6147  024576  012603                            MOV     (SP)+,R3        ,,POP STACK INTO R3
6148  024600  012602                            MOV     (SP)+,R2        ;,POP STACK INTO R2
6149  024602  012601                            MOV     (SP)+,R1        ,;POP STACK INTO R1
6150  024604  012600                            MOV     (SP)+,R0        ;,POP STACK INTO R0
6151  024606  000207                            RTS     PC              ,RETURN
6152
6153                                    ,THIS ROUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST
6154                                    ,CALL
6155                                    ,       MOV     #TESTNUM,$TSTNM  ,LOAD THE TEST NUMBER
6156                                    ,       JSR     PC,LODPRM
6157                                    ,       RETURN
6158
6159  024610                            LODPRM
6160  024610  010146                            MOV     R1,-(SP)        ,,PUSH R1 ON STACK
6161  024612  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
6162  024614  010346                            MOV     R3,-(SP)        ;,PUSH R3 ON STACK
6163  024616  010446                            MOV     R4,-(SP)        ,;PUSH R4 ON STACK
6164  024620  005004                            CLR     R4              ,CLEAR P4
6165  024622  113704  001102                    MOVB    $TSTNM,R4       ;GET THE TEST NUMBER
6166  024626  006304                            ASL     R4              ,SETUP TO ADDRESS WORDS
6167  024630  016401  001536                    MOV     PRMPT(R4),R1    ,GET THE TEST'S PARAMETER TABLE ADDRESS
6168  024634  012702  001504                    MOV     #PRM,R2         ,PARAMETER EXECUTION TABLE
6169  024640  005003                            CLR     R3              ,R3 IS USED AS A COUNTER
6170  024642  013704  001254                    MOV     CHKDRV,R4       ,DRIVE'S ADDRESS
6171  024646  012122                            MOV     (R1)+,(R2)+     ,PARAMETER SPECIFIER
```

```
6172  024650  006237  001504          1$:     ASR     PRM             ;THIS PARAMETER USED IN THE TEST ?
6173  024654  103002                          BCC     2$              ;BR IF NOT
6174  024656  012122                          MOV     (R1)+,(R2)+     ;LOAD THE VALUE
6175  024660  000401                          BR      3$              ;CONTINUE
6176  024662  005022                  2$      CLR     (R2)+           ;CLEAR THE UNUSED PARAMETER LOCATION
6177  024664  005203                  3$.     INC     R3              ;COUNT THE POSITION IN THE OUTPUT TABLE
6178  024666  020327  000014                  CMP     R3,#12          ;FINISHED ?
6179  024672  001430                          BEQ     6$              ;BR IF YES
6180  024674  020327  000003                  CMP     R3,#3           ;DOING THE CYLINDER ADDRESSES ?
6181  024700  001363                          BNE     1$              ;BR IF NOT
6182  024702  132764  000004  034376          BITB    #BIT02,DRVTYP(R4)  ;RP06 ?
6183  024710  001016                          BNE     5$              ;BR IF IT IS
6184  024712  062703  000002                  ADD     #2,R3           ;COUNT THE BYPASSED PARAMETERS (FC' & LC')
6185  024716  006237  001504                  ASR     PRM             ;SHIFT THE COUNTER
6186  024722  103002                          BCC     4$              ;BR IF FC' IS NOT USED
6187  024724  062701  000002                  ADD     #2,R1           ;MOVE THE INPUT POINTER
6188  024730  006237  001504          4$      ASR     PRM             ;COUNT THE PARAMETER
6189  024734  103345                          BCC     1$              ;BR IF LC' NOT USED
6190  024736  062701  000002                  ADD     #2,R1           ;MOVE THE INPUT PINTER
6191  024742  000742                          BR      1$              ;KEEP GOING
6192  024744  000741                          BR      1$              ;KEEP GOING
6193  024746  162702  000004          5$      SUB     #4,R2           ;BACKUP THE OUTPUT POINTER
6194  024752  000736                          BR      1$              ;KEEP GOING
6195  024754                          6$      .
6196  024754  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
6197  024756  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
6198  024760  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
6199  024762  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
6200  024764  000207                          RTS     PC              ;RETURN
6201
6202                                  ;THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
6203                                  ;INTO DPB.B+2 AND DPB C+2, DEPENDING ON THE STATE OF "CONTROL SWITCH"
6204                                  ;BIT07.
6205                                  ;CALL
6206                                  ;       JSR     PC,@#LDCMD
6207                                  ;       RETURN
6208
6209  024766  032737  000200  001220  LDCMD   BIT     #SW07,@#C.SWR   ;DO EXPLICIT SEEKS?
6210  024774  001007                          BNE     1$              ;YES--BRANCH
6211  024776  012737  000173  004126          MOV     #READHD,@#DPB B+2 ;NO--SET UP FOR READ HEADER AND
6212  025004  012737  000173  004146          MOV     #READHD,@#DPB.C+2 ;DATA COMMAND
6213  025012  000406                          BR      2$
6214  025014  012737  000105  004126  1$      MOV     #SEEK,@#DPB.B+2 ;SETUP FOR SEEK COMMAND
6215  025022  012737  000105  004146          MOV     #SEEK,@#DPB.C+2
6216  025030  000207                  2$      RTS     PC
6217
6218                                  ;THIS ROUTINE WILL CALL THE RP04/5/6 DRIVER AND THEN WAIT ON THE FUNCTION
6219                                  ;TO COMPLETE  IF AN ERROR OCCURS IT IS REPORTED
6220                                  ;CALL
6221                                  ;       FILL "DPB" WITH COMMAND INFORMATION
6222                                  ;       JSR     R0,@#CALL A
6223                                  ;       RETURN
6224
6225  025032  005037  001206          CALL A  CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
6226  025036  004037  035302                  JSR     R0,@#RP04       ;CALL RP04 DRIVER
6227  025042  004104                          DPB A
```

```
6228  025044  000772                          BR      CALL.A
6229  025046  005737  004122          1$      TST     @#DPB.A+16      ;DONE?
6230  025052  001775                          BEQ     1$              ;NO--LOOP
6231  025054  100032                          BPL     3$              ;BRANCH IF NO ERROR
6232  025056  012737  025132  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
6233  025064  013737  004116  001270          MOV     @#DPB.A+12,@#CYL.DS ;CYLINDER
6234  025072  113737  004115  001274          MOVB    @#DPB.A+11,@#TRK.DS ;TRACK
6235  025100  113737  004114  001272          MOVB    @#DPB.A+10,@#SEC.DS ;SECTOR
6236  025106  012746  004122                  MOV     #DPB.A+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6237  025112  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
6238  025116  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
6239  025120  104041                          ERROR   41              ;
6240  025122  104042                          ERROR   42              ,PARITY ERROR
6241  025124  104043                          ERROR   43              ,UNSAFE ERROR
6242  025126  104044                          ERROR   44              ;NON-I/O ERROR
6243  025130  104045                          ERROR   45              ;I/O ERROR
6244  025132  013746  004122          2$      MOV     DPB A+16,-(SP)  ,STATUS WORD
6245  025136  004737  026206                  JSR     PC,LOP CK       ,SEE IF LOOP, ABORT, OR CONTINUE
6246  025142  000200                  3$      RTS     R0              ,RETURN
6247
6248                                  ,THIS ROUTINE IS THE SAME AS "CALL A" EXCEPT FOR THE DPB USED AND IF
6249                                  ,THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,
6250                                  ,AND SECTOR) READ IS CHECKED FOR VALIDITY
6251                                  ;CALL
6252                                  ,       FILL DPB
6253                                  ,       JSR     R0,@#CALL.B
6254                                  ,       RETURN
6255
6256  025144  005037  001206  CALL B  CLR     @#$ESCAPE       ,NO ESCAPE ADDRESS
6257  025150  004037  035302          JSR     R0,@#RP04       ,CALL RP04 DRIVER
6258  025154  004124                  DPB B
6259  025156  000772                  BR      CALL B
6260  025160  005737  004142          1$      TST     DPB B+16        ;DONE?
6261  025164  001775                          BEQ     1$              ,NO--BRANCH
6262  025166  100042                          BPL     4$              ;BRANCH IF NO ERROR
6263  025170  012737  025262  001206          MOV     #3$,$ESCAPE     ,,ESCAPE TO 3$ ON ERROR
6264  025176  013737  004136  001270          MOV     @#DPB B+12,@#CYL DS ;CYLINDER
6265  025204  113737  004135  001274          MOVB    @#DPB.B+11,@#TRK DS ,TRACK
6266  025212  113737  004134  001272          MOVB    @#DPB B+10,@#SEC DS ,SECTOR
6267  025220  012746  004142                  MOV     #DPB B+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6268  025224  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
6269  025230  062607                          ADD     (SP)+,PC        ,REPORT PROPER ERROR
6270  025232  104041                          ERROR   41              ,
6271  025234  104042                          ERROR   42              ,PARITY ERROR
6272  025236  104043                          ERROR   43              ,UNSAFE ERROR
6273  025240  104044                          ERROR   44              ,NON-I/O ERROR
6274  025242  005737  004220                  TST     RP REG+RPER1    ,DRIVE ERROR ?
6275  025246  001404                          BEQ     2$              ,BR IF NOT
6276  025250  032737  177677  004220          BIT     # C100,RP REG+RPER1  ,SEE IF ONLY 'HCE' SET
6277  025256  001406                          BEQ     4$              ,BR IF IT IS
6278  025260  104045                  2$      ERROR   45              ,I/O ERROR
6279  025262  013746  004142          3$      MOV     DPB B+16,-(SP)  ,STATUS WORD
6280  025266  004737  026206                  JSR     PC,LOP CK       ,SEE IF LOOP, ABORT, OR CONTINUE
6281  025272  000410                          BR      5$              ,CHECK FOR STALL
6282  025274  123727  004126  000173  4$      CMPB    @#DPB B+2,#READHD ,DOING IMPLIED SEEKS?
6283  025302  001004                          BNE     5$              ,NO--BRANCH
```

```
6284  025304  004037  026526                  JSR     R0,@#VERIFY     ;YES--GO CHECK THE DATA
6285  025310  004134                          DPB B+10
6286  025312  000407                          BR      6$              ;ERROR DURING VERIFY
6287  025314  032737  040000  001220  5$      BIT     #SW14,@#C.SWR   ;STALL?
6288  025322  001403                          BEQ     6$              ;NO--BRANCH
6289  025324  004037  026364                  JSR     R0,@#STALL      ;YES--CALL STALL ROUTINE
6290  025330  001354                           WORD   STALL1          ;STALL TIME POINTER
6291  025332  000200                  6$      RTS     R0              ;RETURN
6292
6293                                          ;THIS ROUTINE IS THE SAME AS "CALL B" EXCEPT FOR THE DPB USED
6294                                          ;CALL
6295                                          ;       FILL DPB
6296                                          ;       JSR     R0,@#CALL C
6297                                          ;       RETURN
6298
6299  025334  005037  001206          CALL.C  CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
6300  025340  004037  035302                  JSR     R0,@#RP04       ;CALL RP04 DRIVER
6301  025344  004144                          DPB C
6302  025346  000772                          BR      CALL.C
6303  025350  005737  004162          1$      TST     @#DPB.C+16      ;DONE?
6304  025354  001775                          BEQ     1$              ;NO--LOOP
6305  025356  100042                          BPL     4$              ;YES--BRANCH IF NO ERROR
6306  025360  012737  025452  001206          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
6307  025366  013737  004156  001270          MOV     @#DPB.C+12,@#CYL.DS ;CYLINDER
6308  025374  113737  004155  001274          MOVB    @#DPB.C+11,@#TRK.DS ;TRACK
6309  025402  113737  004154  001272          MOVB    @#DPB.C+10,@#SEC.DS ;SECTOR
6310  025410  012746  004162                  MOV     #DPB C+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6311  025414  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
6312  025420  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
6313  025422  104041                          ERROR   41              ;
6314  025424  104042                          ERROR   42              ;PARITY ERROR
6315  025426  104043                          ERROR   43              ;UNSAFE ERROR
6316  025430  104044                          ERROR   44              ;NON-I/O ERROR
6317  025432  005737  004220                  TST     RP REG+RPER1    ;DRIVE ERROR ?
6318  025436  001404                          BEQ     2$              ;BR IF NOT
6319  025440  032737  177677  004220          BIT     # C100,RP REG+RPER1  ;SEE IF ONLY 'HCE' SET
6320  025446  001406                          BEQ     4$              ;BR IF IT IS
6321  025450  104045                  2$      ERROR   45              ;I/O ERROR
6322  025452  013746  004162          3$      MOV     DPB.C+16,-(SP)  ;STATUS WORD
6323  025456  004737  026206                  JSR     PC,LOP CK       ;SEE IF LOOP, ABORT, OR CONTINUE
6324  025462  000410                          BR      5$
6325  025464  123727  004146  000173  4$      CMPB    @#DPB C+2,#READHD ;DOING IMPLIED SEEK?
6326  025472  001004                          BNE     5$              ;NO--EXIT
6327  025474  004037  026526                  JSR     R0,@#VERIFY     ;YES--CHECK THE DATA
6328  025500  004154                          DPB C+10
6329  025502  000407                          BR      6$              ;ERROR DURING VERIFY
6330  025504  032737  040000  001220  5$      BIT     #SW14,@#C SWR   ;STALL?
6331  025512  001403                          BEQ     6$              ;NO--BRANCH
6332  025514  004037  026364                  JSR     R0,@#STALL      ;YES--CALL STALL ROUTINE
6333  025520  001354                          .WORD   STALL1          ;STALL TIME POINTER
6334  025522  000200                  6$      RTS     R0
6335
6336
6337                                          ;THIS ROUTINE IS THE SAME AS "CALL A" EXCEPT FOR THE DPB USED AND
6338                                          ;ON AN ERROR LOCATION "ERR CT" IS EXAMINED  IF ERR CT IS EQUAL TO
6339                                          ;$ERFLG EXIT IS TO THE NEXT TEST
```

```
                                        ,CALL
                                        ,       FILL DPB
                                        ;       JSR     R0,@#DRVCAL
                                        ;       RETURN
                                        
025524  005037  001206          DRVCAL. CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
025530  005037  001334                  CLR     @#WCEFLG        ;CLEAR WRITE CHECK ERROR FLAG
025534  004037  035302                  JSR     R0,@#RP04       ;CALL RP04 DRIVER
025540  004164                          DTADPB
025542  000770                          BR      DRVCAL
025544  005737  004202          DRVCL1  TST     @#DTADPB+16     ;DONE
025550  001775                          BEQ     DRVCL1          ;NO--LOOP
025552  100402                          BMI     1$              ;BR IF ERRORS
025554  000137  026166                  JMP     10$             ;NO ERRORS
025560                          1$.
025560  012737  025634  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
025566  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
025574  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
025602  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
025610  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
025614  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
025620  062607                          ADD     (SP)+,PC        ,REPORT PROPER ERROR
025622  104041                          ERROR   41              ;
025624  104042                          ERROR   42              ;PARITY ERROR
025626  104043                          ERROR   43              ;UNSAFE ERROR
025630  104044                          ERROR   44              ;NON-I/O ERROR
025632  104045                          ERROR   45              ,I/O ERROR
025634  122737  000020  001102  2$      CMPB    #20,@#$TSTNM    ;TEST 20?
025642  001137                          BNE     8$              ,NO--BRANCH
025644  013746  004202                  MOV     DTADPB+16,-(SP) ,STATUS WORD
025650  004737  026206                  JSR     PC,LOP CK       ;SEE IF LOOP, ABORT, OR CONTINUE
025654  122737  000151  004166          CMPB    #WRCKD,@#DTADPB+2 ;DOING A WRITE CHECK?
025662  001133                          BNE     12$             ;NO--BRANCH
025664  032737  040000  004214          BIT     #BIT14,@#RP REG+10 ;IS "WCE"=1?
025672  001527                          BEQ     12$             ;NO--BRANCH
025674  032777  000020  153236          BIT     #SW04,@SWR      ;INHIBIT WRITES?
025702  001123                          BNE     12$             ;YES--BRANCH
025704  112737  000161  004166          MOVB    #WRITE,@#DTADPB+2 ,SETUP FOR A WRITE
025712  005037  001206                  CLR     @#$ESCAPE       ,NO ESCAPE ADDRESS
025716  004037  035302                  JSR     R0,@#RP04       ,DO THE WRITE
025722  004164                          DTADPB
025724  000240                          NOP
025726  005737  004202          3$.     TST     @#DTADPB+16     ,DONE?
025732  001775                          BEQ     3$              ,NO--LOOP
025734  100026                          BPL     4$              ,YES--BRANCH IF NO ERROR
025736  012737  026142  001206          MOV     #8$,$ESCAPE     ;;ESCAPE TO 8$ ON ERROR
025744  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ,CYLINDER
025752  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
025760  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC DS ,SECTOR
025766  012746  004202                  MOV     #DTADPB+16,-(SP) ,STATUS/ERROR INDICATOR ADDRESS
025772  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
026000  062607                          ADD     (SP)+,PC        ,REPORT PROPER ERROR
026002  104041                          ERROR   41              ,
026004  104042                          ERROR   42              ,PARITY ERROR
026006  104043                          ERROR   43              ,UNSAFE ERROR
026010  104044                          ERROR   44              ,NON-I/O ERROR
```

```
 6396  026010  104045                          ERROR   45              ;I/O ERROR
 6397  026012  112737  000151  004166  4$.     MOVB    #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
 6398  026020  004037  035302                  JSR     R0,@#RP04       ;DO THE WRITE CHECK
 6399  026024  004164                          DTADPB
 6400  026026  000240                          NOP
 6401  026030  005737  004202          5$      TST     @#DTADPB+16     ;DONE?
 6402  026034  001775                          BEQ     5$              ;NO--LOOP
 6403  026036  100410                          BMI     7$              ;YES--BRANCH IF ERROR
 6404  026040  004037  035302                  JSR     R0,@#RP04       ;DO A 2ND WRITE CHECK
 6405  026044  004164                          DTADPB
 6406  026046  000240                          NOP
 6407  026050  005737  004202          6$      TST     @#DTADPB+16     ;DONE?
 6408  026054  001775                          BEQ     6$              ;NO--LOOP
 6409  026056  100043                          BPL     10$             ;YES--BRANCH IF NO ERROR
 6410  026060  012737  000001  001334  7$      MOV     #1,@#WCEFLG     ;SET THE WRITE CHECK ERROR FLAG
 6411  026066  012737  026142  001206          MOV     #8$,$ESCAPE     ;;ESCAPE TO 8$ ON ERROR
 6412  026074  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
 6413  026102  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
 6414  026110  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
 6415  026116  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
 6416  026122  004737  026246                  JSR     PC,@#ERINDX     ,FORM DISPATCH INDEX
 6417  026126  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
 6418  026130  104041                          ERROR   41              ,
 6419  026132  104042                          ERROR   42              ,PARITY ERROR
 6420  026134  104043                          ERROR   43              ;UNSAFE ERROR
 6421  026136  104044                          ERROR   44              ,NON-I/O ERROR
 6422  026140  104046                          ERROR   46              ,FATAL WRITE CHECK
 6423  026142  013746  004202          8$      MOV     DTADPB+16,-(SP) ,STATUS WORD
 6424  026146  004737  026206                  JSR     PC,LOP.CK       ,SEE IF LOOP, ABORT, OR CONTINUE
 6425  026152  123737  001364  001103  12$     CMPB    @#ERR.CT,@#$ERFLG ;GO TO NEXT TEST?
 6426  026160  101002                          BHI     10$             ,NO--BRANCH
 6427  026162  013700  001252          9$      MOV     @#BYPASS,R0     ,YES--GET EXIT ADDRESS
 6428  026166  032737  040000  001220  10$     BIT     #SW14,@#C SWR   ,STALL?
 6429  026174  001403                          BEQ     11$             ,NO--BRANCH
 6430  026176  004037  026364                  JSR     R0,@#STALL      ,YES--CALL STALL ROUTINE
 6431  026202  001356                           WORD   STALL2          ,STALL TIME POINTER
 6432  026204  000200                  11$     RTS     R0
 6433
 6434                                          ,THIS SUBROUTINE CHECK FOR LOOP, ABORT, OR CONTINUE SWITCHES AFTER
 6435                                          ,ERRORS 41, 42, 43, 44, 45, AND 46
 6436                                          ,CALL
 6437                                          ,       MOV     DTA+16,-(SP)    ,STATUS WORD FROM DPB IN USE
 6438                                          ,       JSR     PC,LOP CK
 6439                                          ,       RETURN
 6440
 6441  026206  032777  001000  152724  LOP CK  BIT     #SW9,@SWR       ,LOOP ON ERROR
 6442  026214  001402                          BEQ     1$              ,BR IF NOT
 6443  026216  000177  152666                  JMP     @$LPERR         ,START AT THE LOOP ADDRESS
 6444  026222  005037  001206          1$      CLR     $ESCAPE         ,CLEAR ERROR ESCAPE FLAG
 6445  026226  032766  072006  000002          BIT     #BIT14!BIT13!BIT12!BIT10!BIT02!BIT01,2(SP)  ,CHECK ERROR TYPE
 6446  026234  001402                          BEQ     2$              ,BR IF DRIVE NOT OFFLINE, UNLOADED, OR
 6447                                                                  ,PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
 6448  026236  000137  017366                  JMP     $EOP             TERMINATE DRIVE
 6449  026242  012616                  2$      MOV     (SP)+,(SP)      ,ADJUST RETURN ADDRESS
 6450  026244  000207                          RTS     PC
 6451
```

```
                        ;THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
                        ;TO THE PROPER ERROR CALL.  THE INDEX IS FORMED BY EXAMINING
                        ;THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB
                        ;INDEX          STATUS/ERROR
                        ;-----          ------------
                        ;     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     #DPB+16,-(SP)   ;ADDRESS OF STATUS/ERROR INDICATOR
                        ;       JSR     PC,@#ERINDX     ;FORM INDEX
                        ;       RETURN                  ;INDEX IS ON THE STACK

026246  010046          ERINDX  MOV     R0,-(SP)        ;SAVE R0
026250  010146                  MOV     R1,-(SP)        ;SAVE R1
026252  016600  000006          MOV     6(SP),R0        ;GET STATUS/ERROR INDICATOR POINTER
026256  011037  001260          MOV     (R0),@#SVSTAT   ;SAVE THE STATUS/ERROR INDICATOR
026262  005001                  CLR     R1              ;START INDEX AT ZERO
026264  032710                  BIT     (PC)+,(R0)      ;FORM INDEX OF 0?
026266  020402                   WORD   BIT13!BIT08!BIT01
026270  001027                  BNE     5$              ;YES--BRANCH
026272  032710                  BIT     (PC)+,(R0)      ;FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
026274  006004                   WORD   BIT11!BIT10!BIT02
026276  001023                  BNE     4$              ;YES--BRANCH
026300  032710                  BIT     (PC)+,(R0)      ;FORM UNSAFE INDEX (4)?
026302  050020                   WORD   BIT14!BIT12!BIT04
026304  001017                  BNE     3$              ;YES--BRANCH
026306  032710                  BIT     (PC)+,(R0)      ;FORM NON-I/O ERROR INDEX (6)?
026310  000050                   WORD   BIT05!BIT03
026312  001013                  BNE     2$              ;YES--BRANCH
026314  032710                  BIT     (PC)+,(R0)      ;FORM I/O ERROR INDEX (10)?
026316  000100                   WORD   BIT06
026320  001007                  BNE     1$              ;YES--BRANCH
026322  032710                  BIT     (PC)+,(R0)      ;SOFTWARE TIMEOUT?
026324  001000                   WORD   BIT09
026326  001410                  BEQ     5$              ;NO--FORM INDEX OF 0
026330  122760  000150  177762  CMPB    #150,-16(R0)    ;YES--I/O?
026336  003001                  BGT     2$              ;NO--BRANCH
026340  005201          1$      INC     R1              ;INDEX=10---ERROR=45 OR 46
026342  005201          2$      INC     R1              ;INDEX=6---ERROR=44
026344  005201          3$      INC     R1              ;INDEX=4---ERROR=43
026346  005201          4$      INC     R1              ;INDEX=2---ERROR=42
026350  006301          5$      ASL     R1              ;INDEX=0---ERROR=41
026352  010166  000006          MOV     R1,6(SP)        ;RETURN INDEX TO USER
026356  012601                  MOV     (SP)+,R1        ;RESTORE R1
026360  012600                  MOV     (SP)+,R0        ;RESTORE R0
026362  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
```

```
6508                                    ,       JSR     R0,@#STALL
6509                                    ,       TIME POINTER            ;WHERE TO FIND THE STALL TIME
6510
6511  026364  013046                    STALL   MOV     @(R0)+,-(SP)    ;PICKUP STALL TIME
6512  026366  032737  020000  001220              BIT     #SW13,@#C SWR   ;USE A RANDOM TIME?
6513  026374  001406                            BEQ     1$              ;NO--BRANCH
6514  026376  004737  023510                    JSR     PC,@#$RAND      ;YES--FORM RANDOM NUMBER
6515  026402  013716  023610                    MOV     @#$LONUM,(SP)   ,AND USE IT FOR THE STALL TIME
6516  026406  042716  177700                    BIC     #^C77,(SP)      ;BUT NEVER > 64 MILLISECONDS
6517  026412  005046                    1$      CLR     -(SP)           ;CLEAR TEMP. LOCATION
6518  026414  162766  000001  000002    2$      SUB     #1,2(SP)        ,MORE STALL REQUIRED?
6519  026422  103407                            BLO     4$              ,NO--BRANCH
6520  026424  012716  000144                    MOV     #100.,(SP)      ,STALL FOR ABOUT 1 MILLISECOND
6521  026430  005700                    3$      TST     R0              ,NOP TO KILL TIME
6522  026432  005366  000000                    DEC     0(SP)           ,COUNT
6523  026436  001374                            BNE     3$              ,LOOP IF MORE COUNTS NEEDED
6524  026440  000765                            BR      2$
6525  026442  022626                    4$      CMP     (SP)+,(SP)+     ,CLEAN OFF THE STACK
6526  026444  000200                            RTS     R0              ,EXIT
6527
6528
6529                                    ,ROUTINE TO PROVIDE A 2 MS STALL AFTER A SEEK OPERATION IN THE SEEK TIMING
6530                                    ,TESTS   THIS STALL IS REQUIRED TO COMPENSATE FOR THE 'ACCESS READY' DELAY
6531                                    ,IN THE RP04   THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES
6532                                    ,CALL
6533                                    ,       JSR     PC,@#TWOMS
6534                                    ,       RETURN
6535
6536  026446  013746  177776            TWO'MS  MOV     @#PS,-(SP)      ,SAVE THE PRESENT PROCESSOR STATUS
6537  026452  012737  000240  177776            MOV     #(5*32.),@#PS   ,SET THE PROCESSOR PRIORITY TO 5
6538  026460  017746  152710                    MOV     @PKV,-(SP)      ,SAVE THE OLD CLOCK VECTOR ADDRESS
6539  026464  012777  026510  152702            MOV     #1$,@PKV        ,SETUP NEW VECTOR ADDRESS
6540  026472  012777  000310  152702            MOV     #200.,@PKB      ,LOAD THE CLOCK BUFFER
6541  026500  012777  000101  152672            MOV     #101,@PKCS      ,START THE CLOCK
6542  026506  000001                            WAIT                    ,WAIT FOR 2 MS
6543  026510  062706  000004            1$      ADD     #4,SP           ,INCREMENT STACK FOR RETURN
6544  026514  012677  152654                    MOV     (SP)+,@PKV      ,RESTORE OLD CLOCK VECTOR
6545  026520  012637  177776                    MOV     (SP)+,@#PS      ,RESTORE THE OLD PROCESSOR STATUS
6546  026524  000207                            RTS     PC              ,RETURN
6547
6548                                    ,ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS
6549                                    ,CALL
6550                                    ,       JSR     R0,@#VERIFY
6551                                    ,       ADR POINTER             ,ADDRESS OF DPB+10 (SECTOR NUMBER)
6552                                    ,       RETURN
6553
6554  026526  010146                    VERIFY  MOV     R1,-(SP)        ;SAVE R1
6555  026530  012001                            MOV     (R0)+,R1        ;GET ADDRESS OF DPB+10
6556  026532  042737  010000  047712            BIC     #FMT22,@#BUFFER ;STRIP FORMAT BIT FROM CYLINDER NUMBER
6557  026540  023761  047712  000002            CMP     @#BUFFER,2(R1)  ;CYLINDER NUMBER OK?
6558  026546  001003                            BNE     1$              ;NO--BRANCH
6559  026550  023711  047714                    CMP     @#BUFFER+2,(R1) ;YES--HOW ABOUT TRACK/SECTOR?
6560  026554  001441                            BEQ     3$              ,BRANCH IF GOOD
6561  026556  013737  047712  001262    1$      MOV     @#BUFFER,@#CYL RD ,SAVE THE EXPECTED AND THE
6562  026564  113737  047715  001264            MOVB    @#BUFFER+3,@#TRK RD ;RECIEVED CYLINDER, TRACK,
6563  026572  113737  047714  001266            MOVB    @#BUFFER+2,@#SEC RD ,AND SECTOR
```

```
6564  026600  112137  001272                  MOVB    (R1)+,@#SEC DS
6565  026604  112137  001274                  MOVB    (R1)+,@#TRK DS
6566  026610  011137  001270                  MOV     (R1),@#CYL DS
6567  026614  012737  026626  001206          MOV     #2$,$ESCAPE     ;ESCAPE TO 2$ ON ERROR
6568  026622  005740                          TST     -(R0)           ;MAKE IT TEST PC+4
6569  026624  104012                          ERROR   12              ;REPORT THE ERROR
6570  026626  012737  000107  004106  2$      MOV     #RECAL,DPB A+2  ;LOAD RECALIBRATE ORDER CODE
6571  026634  004037  025032                  JSR     R0,@#CALL A     ;GO EXECUTE THE COMMAND
6572  026640  005037  001206                  CLR     $ESCAPE         ;CLEAR ERROR ESCAPE FLAG
6573  026644  032777  001000  152266          BIT     #SW9,@SWR       ;LOOP ON ERROR ?
6574  026652  001404                          BEQ     4$              ;BR IF NOT
6575  026654  000177  152230                  JMP     @$LPERR         ;RETURN TO ERROR LOOP ADDRESS
6576  026660  062700  000002          3$      ADD     #2,R0           ;INCREMENT RETURN ADDRESS
6577  026664  012601                  4$      MOV     (SP)+,R1        ;RESTORE R1
6578  026666  000200                          RTS     R0              ;EXIT
6579
6580                                  ;THIS ROUTINE WILL PERFORM A "MASSBUS" INIT  FOLLOWED BY
6581                                  ;A "RECALIBRATE" ON THE DRIVE UNDER TEST
6582                                  ;NOTE  THIS ROUTINE DESTROYS R1 AND R4
6583                                  ;CALL
6584                                  ;       JSR     R0,SRCH00       ;DO A MASSBUS INIT  AND RECAL
6585                                  ;       RETURN1                 ;RETURN HERE IF NO ERROR
6586                                  ;       RETURN2                 ;RETURN HERE ON ERROR
6587
6588  026670  005001                  SRCH00  CLR     R1              ;INCASE OF ERROR (TYPTIM)
6589  026672  005037  177776                  CLR     @#PS
6590  026676  012777  037376  005612          MOV     #ISR,@RPVEC     ;SETUP INTERRUPT VECTOR
6591  026704  013704  034514                  MOV     @#RPADR,R4      ;PICKUP ADDRESS OF RPCS1
6592  026710  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT
6593  026716  005037  004174                  CLR     @#DTADPB+10     ;TRACK=0, SECTOR=0
6594  026722  005037  004176                  CLR     @#DTADPB+12     ;CYLINDER =0
6595  026726  012737  000107  004166          MOV     #RECAL,@#DTADPB+2 ;COMMAND = RECALIBRATE
6596  026734  005037  001206                  CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
6597  026740  004037  035302                  JSR     R0,@#RP04       ;CALL THE DRIVER
6598  026744  004164                          DTADPB                  ;DPB POINTER
6599  026746  000440                          BR      4$              ;QUEUE IS FULL
6600  026750  005737  004202          1$      TST     DTADPB+16       ;WAIT ON DONE
6601  026754  001775                          BEQ     1$
6602  026756  100030                          BPL     3$              ;TAKE NORMAL EXIT IF NO ERROR
6603  026760  012737  027034  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
6604  026766  013737  004176  001270          MOV     @#DTADPB+12,@#CYL DS ;CYLINDER
6605  026774  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK DS ;TRACK
6606  027002  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC DS ;SECTOR
6607  027010  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6608  027014  004737  026246                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
6609  027020  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
6610  027022  104041                          ERROR   41              ;
6611  027024  104042                          ERROR   42              ;PARITY ERROR
6612  027026  104043                          ERROR   43              ;UNSAFE ERROR
6613  027030  104044                          ERROR   44              ;NON-I/O ERROR
6614  027032  104045                          ERROR   45              ;I/O ERROR
6615  027034  005720                  2$      TST     (R0)+           ;ADJUST FOR ERROR EXIT
6616  027036  000404                          BR      4$              ;GO TO THE EXIT
6617  027040  005064  000006          3$      CLR     RPDA(R4)        ;TRACK AND SECTOR = 0
6618  027044  005064  000034                  CLR     RPCA(R4)        ;CYLINDER = 0
6619  027050  000200                  4$      RTS     R0              ;RETURN
```

```
6620
6621                                    ;THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST
6622
6623  027052  000002                    DORTI   RTI                     ;RETURN FROM INTERRUPT
6624
6625                                    ;THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE "TIMING ROUTI ES
6626                                    ;CALL
6627                                    ;       JSR     PC,@#STRTMR
6628                                    ;       RETURN
6629
6630  027054  104412                    STRTMR  SAVREG                  ;SAVE R0-R5
6631  027056  012700  001276                    MOV     #TIM UP,R0      ;START AT TIM UP (MINIMUM)
6632  027062  012701  001332                    MOV     #TIM PT,R1      ;STOP AT TIM PT
6633  027066  005020                    1$      CLR     (R0)+           ;CLEAR
6634  027070  020001                            CMP     R0,R1           ;DONE?
6635  027072  103775                            BLO     1$              ;NO--BRANCH
6636  027074  012710  047712                    MOV     #BUFFER,(R0)    ;SETUP POINTER
6637  027100  012737  077777  001276            MOV     # CBIT15,@#TIM UP ;SET MINIMUM TIME TO MAXIMUM
6638  027106  012737  077777  001314            MOV     # CBIT15,@#TIM DN ;POSITIVE NUMBER
6639  027114  104413                            RESREG                  ;RESTORE R0-R5
6640  027116  000207                            RTS     PC              ;RETURN
6641
6642                                    ;THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND
6643                                    ;MAINTAIN THE MINIMUM AND MAXIMUM TIMES
6644                                    ;NOTE  THIS ROUTINE DESTROYS R2
6645                                    ;CALL
6646                                    ;       MOV     #TP,R3          ;PARAMETER POINTER
6647                                    ;       MOV     FLAG,R5         ;FLAG=0=COUNT UP
6648                                    ;                               ;FLAG=-1=COUNT DOWN
6649                                    ;       JSR     PC,@#COUNT
6650                                    ;       RETURN
6651
6652  027120  012702  001276            COUNT   MOV     #TIM UP,R2      ;PICKUP THE "UP" POINTER
6653  027124  005705                            TST     R5              ;USE IT?
6654  027126  001402                            BEQ     1$              ;YES--BRANCH
6655  027130  012702  001314                    MOV     #TIM DN,R2      ;NO--PICKUP "DOWN" POINTER
6656  027134  027722  152244            1$      CMP     @PKC,(R2)+      ;LESS THAN PREVIOUS LOW?
6657  027140  002003                            BGE     2$              ;NO--BRANCH
6658  027142  017762  152236  177776            MOV     @PKC,-2(R2)     ;YES--SAVE IT
6659  027150  027763  152230  000004    2$      CMP     @PKC,4(R3)      ;LESS THAN THE LOW LIMIT?
6660  027156  002001                            BGE     3$              ;NO--BRANCH
6661  027160  005212                            INC     (R2)            ;YES--COUNT IT
6662  027162  005722                    3$      TST     (R2)+           ;ADVANCE THE POINTER
6663  027164  027722  152214                    CMP     @PKC,(R2)+      ;GREATER THAN PREVIOUS HIGH?
6664  027170  003403                            BLE     4$              ;NO--BRANCH
6665  027172  017762  152206  177776            MOV     @PKC,-2(R2)     ;YES--SAVE IT
6666  027200  027763  152200  000006    4$      CMP     @PKC,6(R3)      ;GREATER THAN THE HIGH LIMIT?
6667  027206  003401                            BLE     5$              ;NO--BRANCH
6668  027210  005212                            INC     (R2)            ;YES--COUNT IT
6669  027212  005722                    5$      TST     (R2)+           ;ADVANCE THE POINTER
6670  027214  067722  152164                    ADD     @PKC,(R2)+      ;ADD THIS COUNT TO THE TOTAL
6671  027220  005522                            ADC     (R2)+
6672  027222  005212                            INC     (R2)            ;COUNT THIS READING
6673  027224  022737  056202  001332            CMP     #BUFFER+<4*814 >,@#TIM PT ;SAVE THIS COUNT?
6674  027232  101406                            BLOS    6$              ;NO--BRANCH
6675  027234  017777  152144  152070            MOV     @PKC,@TIM PT    ;YES--WELL SAVE IT THEN
```

```
6676  027242  062737  000002  001332          ADD     #2,@#TIM.PT     ,ADVANCE THE POINTER
6677  027250  000207                  6$      RTS     PC              ;RETURN
6678
6679                                  ,THIS ROUTINE IS USED TO TYPE THE MINIMUM,
6680                                  ,MAXIMUM, AND AVERAGE TIMES FOR THE TIMING TESTS
6681                                  ,IT WILL ALSO CHECK THE TIMES TO ENSURE
6682                                  ,THEY ARE WITHIN TOLERANCE AND IF NOT FLAG THE BAD TIMES
6683                                  ,NOTE  THIS ROUTINE DESTROYS R2-R5
6684                                  ,CALL
6685                                  ,       JSR     R0,@#TYPTIM     ,GO REPORT THE TIMES
6686                                  ,       TABLE                   ;POINT TO THE PROPER TABLE
6687                                  ,       RETURN
6688                                  ,
6689                                  ,TABLE MSGADR1                  ;ADDRESS OF ASCIZ MESSAGE NUMBER 1
6690                                  ,       MSGADR2                 ,ADDRESS OF ASCIZ MESSAGE NUMBER 2
6691                                  ,       MIN ALLOWED             ,MINIMUM TIME ALLOWED
6692                                  ,       MAX ALLOWED             ,MAXIMUM TIME ALLOWED
6693
6694  027252  012002                  TYPTIM  MOV     (R0)+,R2        ,PICKUP THE TABLE POINTER
6695  027254  032777  000100  151656          BIT     #SW06,@SWR      ,INHIBIT TIME REPORTS?
6696  027262  001145                          BNE     7$              ,YES--BRANCH
6697  027264  012237  027304                  MOV     (R2)+,2$        ,ADDRESS OF MESSAGE NUMBER 1
6698  027270  012205                          MOV     (R2)+,R5        ,ADDRESS OF MESSAGE NUMBER 2
6699  027272  012203                          MOV     (R2)+,R3        ,PICKUP THE LOW LIMIT
6700  027274  011202                          MOV     (R2),R2         ,AND THE HIGH LIMIT
6701  027276  012704  001276                  MOV     #TIM UP,R4      ,PARAMETER POINTER
6702  027302  104401                  1$      TYPE                    ,TYPE THE MESSAGE
6703  027304  000000                  2$       WORD   0               ,ASCIZ MESSAGE POINTER GOES HERE
6704  027306  005764  000014                  TST     14(R4)          ,DID ANY COUNTS OCCUR?
6705  027312  001527                          BEQ     6$              ,NO--BRANCH
6706  027314  104401  043700                  TYPE    ,MSGMIN         ,"MIN="
6707  027320  012446                          MOV     (R4)+,-(SP)             ,PUT (R4)+ ON THE STACK
6708  027322  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE (R4)+ TO DECIMAL ASCIZ
6709  027326  004737  023450                  JSR     PC,@#$SUPRS     ,TYPE WITHOUT LEADING ZEROS
6710  027332  104401  043725                  TYPE    ,MSGOUS         ,"0 US"
6711  027336  005724                          TST     (R4)+           ,ANY SEEKS BELOW THE LOW LIMIT
6712  027340  001421                          BEQ     3$              ,NO--BRANCH
6713  027342  104401  044040                  TYPE    ,MSG SP         ,"  "
6714  027346  016446  177776                  MOV     -2(R4),-(SP)            ,PUT -2(R4) ON THE STACK
6715  027352  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE -2(R4) TO DECIMAL ASCIZ
6716  027356  004737  023450                  JSR     PC,@#$SUPRS     ,TYPE WITHOUT LEADING ZEROS
6717  027362  104401  043732                  TYPE    ,MBELOW         ,"BELOW THE MINIMUM OF"
6718  027366  010346                          MOV     R3,-(SP)                ;PUT R3 ON THE STACK
6719  027370  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE R3 TO DECIMAL ASCIZ
6720  027374  004737  023450                  JSR     PC,@#$SUPRS     ,TYPE WITHOUT LEADING ZEROS
6721  027400  104401  043725                  TYPE    ,MSGOUS
6722  027404  104401  043707          3$      TYPE    ,MSGMAX         ,"MAX="
6723  027410  012446                          MOV     (R4)+,-(SP)             ;PUT (R4)+ ON THE STACK
6724  027412  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE (R4)+ TO DECIMAL ASCIZ
6725  027416  004737  023450                  JSR     PC,@#$SUPRS     ,TYPE WITHOUT LEADING ZEROS
6726  027422  104401  043725                  TYPE    ,MSGOUS
6727  027426  005724                          TST     (R4)+           ,ANY SEEKS ABOVE THE HIGH LIMIT
6728  027430  001421                          BEQ     4$              ,NO--BRANCH
6729  027432  104401  044040                  TYPE    ,MSG.SP         ,YES--REPORT HOW MANY
6730  027436  016446  177776                  MOV     -2(R4),-(SP)            ,PUT -2(R4) ON THE STACK
6731  027442  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE -2(R4) TO DECIMAL ASCIZ
```

```
6732  027446  004737  023450          JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6733  027452  104401  043761          TYPE    ,MABOVE         ;"ABOVE THE MAXIMUM OF"
6734  027456  010246                  MOV     R2,-(SP)                ;PUT R2 ON THE STACK
6735  027460  004737  023220          JSR     PC,@#$SB2D      ;CHANGE R2 TO DECIMPL ASCIZ
6736  027464  004737  023450          JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6737  027470  104401  043725          TYPE    ,MSGOUS
6738  027474  104401  043716  4$      TYPE    ,MSGAVG         ;"AVG="
6739  027500  012446                  MOV     (R4)+,-(SP)     ;FORM THE AVERAGE
6740  027502  012446                  MOV     (R4)+,-(SP)
6741  027504  012446                  MOV     (R4)+,-(SP)
6742  027506  004737  023612          JSR     PC,@#$DIV
6743  027512  006126                  ROL     (SP)+           ;IS THE REMAINDER OVER HALF?
6744  027514  100001                  BPL     5$              ;NO--BRANCH
6745  027516  005216                  INC     (SP)            ;YES--ROUND UP
6746  027520                  5$
6747  027520  004737  023220          JSR     PC,@#$SB2D      ;CHANGE  TO DECIMAL ASCIZ
6748  027524  004737  023450          JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6749  027530  104401  043725          TYPE    ,MSGOUS
6750  027534  104401  044040          TYPE    ,MSG SP
6751  027540  016446  177776          MOV     -2(R4),-(SP)            ;PUT -2(R4) ON THE STACK
6752  027544  004737  023220          JSR     PC,@#$SB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
6753  027550  004737  023450          JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6754  027554  104401  044010          TYPE    ,MSGNUM         ;"SEEKS TIMED"
6755  027560  010537  027304          MOV     R5,2$           ;NEXT MESSAGE POINTER
6756  027564  001404                  BEQ     7$              ;IF NONE EXIT
6757  027566  005005                  CLR     R5              ;NO MORE THAN 2
6758  027570  000644                  BR      1$
6759  027572  104401  044025  6$      TYPE    ,MSGNON
6760  027576  000200          7$      RTS     R0              ;EXIT
6761
6762                          ;THIS SUBROUTINE WILL INCREMENT THE TRACK
6763                          ;NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'
6764                          ;CALL
6765                          ;       JSR     R0,@#INCTRK
6766                          ;       RETURN1                 ;TRACK NUMBER GREATER THAN LT15
6767                          ;       RETURN2                 ;TRACK NUMBER INCREMENTED
6768
6769  027600  020237  001520  INCTRK  CMP     R2,@#LT         ;LAST TRACK COMPLETED?
6770  027604  001410                  BEQ     2$              ;YES--EXIT
6771  027606  063702  001522          ADD     @#IT,R2         ;NO--UPDATE TRACK
6772  027612  020237  001520          CMP     R2,@#LT         ;TRACK TO BIG?
6773  027616  003402                  BLE     1$              ;NO--EXIT
6774  027620  013702  001520          MOV     @#LT,R2         ;YES--SET TRACK TO LAST TRACK
6775  027624  005720          1$      TST     (R0)+           ;ADJUST FOR RETURN 2
6776  027626  000200          2$      RTS     R0              ;RETURN
6777
6778
6779                          ;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
6780                          ;NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'
6781                          ;CALL
6782                          ;       JSR     R0,@#INCCYL
6783                          ;       RETURN1                 ;CYLINDER NUMBER GREATER THAN LC15
6784                          ;       RETURN2                 ;CYLINDER NUMBER INCREMENTED
6785
6786  027630  020137  001512  INCCYL  CMP     R1,@#LC         ;LAST CYLINDER COMPLETED?
6787  027634  001410                  BEQ     2$              ;YES--EXIT
```

```
6788  027636  063701  001514                ADD     @#IC,R1          ;NO--UPDATE CYLINDER
6789  027642  020137  001512                CMP     R1,@#LC          ;CYLINDER TO BIG?
6790  027646  003402                        BLE     1$               ,NO--EXIT
6791  027650  013701  001512                MOV     @#LC,R1          ;YES--SET CYLINDER TO LAST CYLINDER
6792  027654  005720              1$        TST     (R0)+            ,ADJUST FOR RETURN 2
6793  027656  000200              2$        RTS     R0               ;RETURN
6794
6795                                ,THIS ROUTINE DECREMENTS THE SECTOR ADDRESS.
6796                                ,CALL
6797                                ,       CLR     -(SP)            ;CLEAR THE STACK
6798                                ,       JSR     PC,DECSEC        ,SUBROUTINE ENTRY
6799                                ,       RETURN
6800
6801  027660  113766  004212  000002  DECSEC  MOVB    RP REG+RPDA,2(SP)  ,PUT THE SECTOR ADDRESS ON THE STACK
6802  027666  005366  000002                DEC     2(SP)            ,DECREMENT THE ADDRESS
6803  027672  100003                        BPL     1$               ,BR IF NOT CORRECTION NEEDED
6804  027674  013766  001634  000002        MOV     PRMLMT+22 ,2(SP)  ,OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
6805  027702  000207              1$        RTS     PC               ,RETURN
6806
6807                                ,THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
6808                                ,WITH ADDRESSES FROM 0 TO 21 WITH EACH ADDRESS
6809                                ,BEING STORED IN 256 CONSECUTIVE LOCATIONS
6810                                ,CALL
6811                                ,       JSR     PC,@#FILBUF
6812                                ,       RETURN
6813
6814  027704  104412              FILBUF  SAVREG                   ,SAVE R0 - R5
6815  027706  005000                      CLR     R0               ,FIRST DISK ADDRESS
6816  027710  012701  047712              MOV     #BUFFER,R1       ,START FILLING HERE
6817  027714  012702  000400        1$    MOV     #256 ,R2         ,DO 256 WORDS
6818  027720  010021              2$      MOV     R0,(R1)+         ,STORE
6819  027722  005302                      DEC     R2               ;MORE?
6820  027724  003375                      BGT     2$               ,YES--BRANCH
6821  027726  005200                      INC     R0               ,NO--UPDATE DISK ADDRESS
6822  027730  023700  001630              CMP     PRMLMT+22,R0     ,DONE?
6823  027734  103367                      BHIS    1$               ,NO--BRANCH
6824  027736  104413                      RESREG                   ,RESTORE R0 - R5
6825  027740  000207                      RTS     PC               ,RETURN
6826
6827                                ,THIS ROUTINE WILL CLEAR THE BUFFER BY
6828                                ,SETTING EACH WORD TO "177400".
6829                                ,CALL
6830                                ,       JSR     R0,@#CLRBUF
6831                                ;       RETURN
6832
6833  027742  104412              CLRBUF  SAVREG                   ;SAVE R0 - R5
6834  027744  012701  177400              MOV     #177400,R1       ,WORD TO FILL BUFFER WITH
6835  027750  012702  047712              MOV     #BUFFER,R2       ,FIRST ADDRESS OF BUFFER
6836  027754  012703  075712              MOV     #BUFFER+<512 *22 >,R3 ;LAST ADDRESS+2 OF BUFFER
6837  027760  010122              1$      MOV     R1,(R2)+         ;FILL WORDS 1, 9,...249,...5625
6838  027762  010122                      MOV     R1,(R2)+         ,FILL WORDS 2,10,...250,...5626
6839  027764  010122                      MOV     R1,(R2)+         ,FILL WORDS 3,11,...251,...5627
6840  027766  010122                      MOV     R1,(R2)+         ,FILL WORDS 4,12,...252,...5628
6841  027770  010122                      MOV     R1,(R2)+         ;FILL WORDS 5,13,...253,...5629
6842  027772  010122                      MOV     R1,(R2)+         ,FILL WORDS 6,14, ..254,...5630
6843  027774  010122                      MOV     R1,(R2)+         ,FILL WORDS 7,15, ..255,...5631
```

```
 6844  027776  010122                    MOV     R1,(R2)+        ;FILL WORDS 8,16,   256,   5632
 6845  030000  020203                    CMP     R2,R3           ;DONE?
 6846  030002  103766                    BLO     1$              ,NO--BRANCH
 6847  030004  104413                    RESREG                  ;RESTORE R0 - R5
 6848  030006  000200                    RTS     R0              ;RETURN FROM CALL
 6849
 6850                            ;THIS ROUTINE IS USED TO CHECK THE DATA BUFFER
 6851                            ,FOR ADDRESSES 0 THROUGH 21 WITH EACH ADDRESS
 6852                            ,BEING STORED IN 256 CONSECUTIVE LOCATIONS
 6853                            ,CALL
 6854                            ,       JSR     R0,@#CKSCTR
 6855                            ,       RETURN
 6856
 6857  030010  104412            CKSCTR  SAVREG                  ;SAVE R0 - R5
 6858  030012  162706  000004            SUB     #4,SP           ;RESERVE TEMP. STORAGE AREA
 6859  030016  005001                    CLR     R1              ;FIRST SECTOR
 6860  030020  012716  047712            MOV     #BUFFER,(SP)    ,FIRST ADDRESS OF DATA BUFFER
 6861  030024  005066  000002            CLR     2(SP)           ;NO ERRORS
 6862  030030  012702  000020    1$      MOV     #16 ,R2 ,LOOP COUNT (16*16=256)
 6863  030034  011603                    MOV     (SP),R3         ;GET 1ST ADDRESS OF THIS SECTORS DATA
 6864  030036                    2$
 6865  030036  020123                    CMP     R1,(R3)+        ;WORD 1
 6866  030040  001063                    BNE     7$              ,BRANCH IF BAD
 6867  030042  020123                    CMP     R1,(R3)+        ;WORD 2
 6868  030044  001061                    BNE     7$              ;BRANCH IF BAD
 6869  030046  020123                    CMP     R1,(R3)+        ,WORD 3
 6870  030050  001057                    BNE     7$              ;BRANCH IF BAD
 6871  030052  020123                    CMP     R1,(R3)+        ,WORD 4
 6872  030054  001055                    BNE     7$              ,BRANCH IF BAD
 6873  030056  020123                    CMP     R1,(R3)+        ,WORD 5
 6874  030060  001053                    BNE     7$              ,BRANCH IF BAD
 6875  030062  020123                    CMP     R1,(R3)+        ,WORD 6
 6876  030064  001051                    BNE     7$              ;BRANCH IF BAD
 6877  030066  020123                    CMP     R1,(R3)+        ,WORD 7
 6878  030070  001047                    BNE     7$              ,BRANCH IF BAD
 6879  030072  020123                    CMP     R1,(R3)+        ;WORD 8
 6880  030074  001045                    BNE     7$              ,BRANCH IF BAD
 6881  030076  020123                    CMP     R1,(R3)+        ;WORD 9
 6882  030100  001043                    BNE     7$              ,BRANCH IF BAD
 6883  030102  020123                    CMP     R1,(R3)+        ;WORD 10
 6884  030104  001041                    BNE     7$              ;BRANCH IF BAD
 6885  030106  020123                    CMP     R1,(R3)+        ;WORD 11
 6886  030110  001037                    BNE     7$              ;BRANCH IF BAD
 6887  030112  020123                    CMP     R1,(R3)+        ;WORD 12
 6888  030114  001035                    BNE     7$              ,BRANCH IF BAD
 6889  030116  020123                    CMP     R1,(R3)+        ;WORD 13
 6890  030120  001033                    BNE     7$              ,BRANCH IF BAD
 6891  030122  020123                    CMP     R1,(R3)+        ,WORD 14
 6892  030124  001031                    BNE     7$              ,BRANCH IF BAD
 6893  030126  020123                    CMP     R1,(R3)+        ,WORD 15
 6894  030130  001027                    BNE     7$              ;BRANCH IF BAD
 6895  030132  020123                    CMP     R1,(R3)+        ;WORD 16
 6896  030134  001025                    BNE     7$              ;BRANCH IF BAD
 6897  030136  005302                    DEC     R2              ,FINISHED WITH THIS SECTORS DATA?
 6898  030140  001336                    BNE     2$              ,NO--BRANCH
 6899  030142  062716  001000    3$      ADD     #512 ,(SP)      ,YES--FIRST ADDRESS OF NEXT SECTOR
```

```
6900  030146  005201                          INC     R1                  ,MOVE TO NEXT SECTOR
6901  030150  023701  001630                  CMP     PRMLMT+22,R1        ;DONE?
6902  030154  103325                          BHIS    1$                  ;NO--BRANCH
6903  030156  005766  000002          4$      TST     2(SP)               ;ERROR OCCUR?
6904  030162  001406                          BEQ     6$                  ;NO--BRANCH
6905  030164  123737  001364  001103          CMPB    @#ERR.CT,@#$ERFLG ;MAX. ERROR OCCURRED?
6906  030172  101002                          BHI     6$                  ;NO--BRANCH
6907  030174  013700  001252          5$      MOV     @#BYPASS,R0         ;TAKE ERROR EXIT
6908  030200  062706  000004          6$      ADD     #4,SP               ;FREE TEMP. AREA
6909  030204  104413                          RESREG                      ;RESTORE R0 - R5
6910  030206  000200                          RTS     R0                  ;RETURN FROM CALL
6911  030210  010304          7$      MOV     R3,R4               ;FORM WORD NUMBER AND
6912  030212  161604                          SUB     (SP),R4             ,ADDRESS TO CONTINUE FROM
6913  030214  010405                          MOV     R4,R5
6914  030216  006204                          ASR     R4                  ,WORD NUMBER
6915  030220  042705  177740                  BIC     # C37,R5
6916  030224  001002                          BNE     8$                  ,BRANCH IF NOT A MULTIPLE OF 16
6917  030226  012705  000040                  MOV     #40,R5              ,SET TO WORD 16
6918  030232  006305          8$      ASL     R5
6919  030234  062705  030036                  ADD     #2$,R5              ;ADDRESS
6920  030240  016337  177776  001126          MOV     -2(R3),@#$BDDAT ,SAVE BAD DATA
6921  030246  005766  000002                  TST     2(SP)               ,FIRST ERROR?
6922  030252  001015                          BNE     10$                 ;NO--BRANCH
6923  030254  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER NUMBER
6924  030262  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK NUMBER
6925  030270  012737  030300  001206          MOV     #9$,$ESCAPE         ;,ESCAPE TO 9$ ON ERROR
6926  030276  104021                          ERROR   21                  ;REPORT THE ERROR
6927  030300  105166  000002          9$      COMB    2(SP)               ,SET ERROR SWITCH
6928  030304  000404                          BR      11$
6929  030306                          10$
6930  030306  012737  030316  001206          MOV     #11$,$ESCAPE        ,,ESCAPE TO 11$ ON ERROR
6931  030314  104022                          ERROR   22                  ;REPORT THE ERROR
6932  030316  032777  001000  150614  11$     BIT     #SW09,@SWR          ;LOOP ON ERROR?
6933  030324  001323                          BNE     5$                  ,YES
6934  030326  032777  000002  150604          BIT     #SW01,@SWR          ;STOP DATA COMPARE?
6935  030334  001310                          BNE     4$                  ,YES--BRANCH
6936  030336  123737  001364  001103          CMPB    @#ERR CT,@#$ERFLG ;MAX. ERRORS?
6937  030344  101713                          BLOS    5$                  ;YES--BRANCH
6938  030346  032777  000040  150564          BIT     #SW05,@SWR          ,REPORT ONLY 1ST ERROR PER SECTOR?
6939  030354  001272                          BNE     3$                  ,YES--BRANCH
6940  030356  000115                          JMP     (R5)
6941
6942                                  ,THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
6943                                  ,DESIRED PATTERN INTO THE DATA BUFFER
6944                                  ,CALL
6945                                  ,       MOV     #NX,R0              ,PATTERN NUMBER INDEX TO R0
6946                                  ,       JSR     PC,@#SETBUF
6947
6948  030360  104412                  SETBUF  SAVREG                      ,SAVE R0 - R5
6949  030362  012701  047712                  MOV     #BUFFER,R1          ,FIRST ADDRESS
6950  030366  013702  004170                  MOV     @#DTADPB+4,R2       ,WORD COUNT
6951  030372  016003  003044          1$      MOV     PAT PT(R0),R3       ;PICKUP PATTERN POINTER
6952  030376  012321                          MOV     (R3)+,(R1)+         ,MOVE WORD 1 INTO DATA BUFFER
6953  030400  012321                          MOV     (R3)+,(R1)+         ,MOVE WORD 2 INTO DATA BUFFER
6954  030402  012321                          MOV     (R3)+,(R1)+         ,MOVE WORD 3 INTO DATA BUFFER
6955  030404  012321                          MOV     (R3)+,(R1)+         ,MOVE WORD 4 INTO DATA BUFFER
```

```
 6956  030406  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 5 INTO DATA BUFFER
 6957  030410  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 6 INTO DATA BUFFER
 6958  030412  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 7 INTO DATA BUFFER
 6959  030414  012321                    MOV     (R3)+,(R1)+     ,MOVE WORD 8 INTO DATA BUFFER
 6960  030416  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 9 INTO DATA BUFFER
 6961  030420  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 10 INTO DATA BUFFER
 6962  030422  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 11 INTO DATA BUFFER
 6963  030424  012321                    MOV     (R3)+,(R1)+     ,MOVE WORD 12 INTO DATA BUFFER
 6964  030426  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 13 INTO DATA BUFFER
 6965  030430  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 14 INTO DATA BUFFER
 6966  030432  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 15 INTO DATA BUFFER
 6967  030434  012321                    MOV     (R3)+,(R1)+     ;MOVE WORD 16 INTO DATA BUFFER
 6968  030436  062702  000020            ADD     #16 ,R2 ,DONE?
 6969  030442  001353                    BNE     1$              ,NO--BRANCH
 6970  030444  104413                    RESREG                  ,RESTORE R0 - R5
 6971  030446  000207                    RTS     PC              ,RETURN
 6972
 6973                                ,THIS ROUTINE COMPARES A 16 WORD DATA PATTERN
 6974                                ,AGAINST THE DATA BUFFER
 6975                                ,CALL
 6976                                ,       MOV     #NX,R0          ,PATTERN NUMBER INDEX TO R0
 6977                                ,       JSR     PC,@#DATCMP
 6978                                ,       RETURN
 6979
 6980  030450  104412                DATCMP  SAVREG                  ;SAVE R0 - R5
 6981  030452  012701  047712                MOV     #BUFFER,R1      ,FIRST ADDRESS OF BUFFER
 6982  030456  013702  004170                MOV     @#DTADPB+4,R2   ,WORD COUNT
 6983  030462  005046                        CLR     -(SP)           ,NO ERROR
 6984  030464  016003  003044        1$      MOV     PAT PT(R0),R3   ,PATTERN POINTER
 6985  030470                        2$
 6986  030470  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 1
 6987  030472  001044                        BNE     4$              ;BRANCH IF DIFFERENT
 6988  030474  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 2
 6989  030476  001042                        BNE     4$              ,BRANCH IF DIFFERENT
 6990  030500  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 3
 6991  030502  001040                        BNE     4$              ,BRANCH IF DIFFERENT
 6992  030504  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 4
 6993  030506  001036                        BNE     4$              ;BRANCH IF DIFFERENT
 6994  030510  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 5
 6995  030512  001034                        BNE     4$              ,BRANCH IF DIFFERENT
 6996  030514  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 6
 6997  030516  001032                        BNE     4$              ,BRANCH IF DIFFERENT
 6998  030520  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 7
 6999  030522  001030                        BNE     4$              ,BRANCH IF DIFFERENT
 7000  030524  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 8
 7001  030526  001026                        BNE     4$              ,BRANCH IF DIFFERENT
 7002  030530  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 9
 7003  030532  001024                        BNE     4$              ,BRANCH IF DIFFERENT
 7004  030534  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 10
 7005  030536  001022                        BNE     4$              ,BRANCH IF DIFFERENT
 7006  030540  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 11
 7007  030542  001020                        BNE     4$              ,BRANCH IF DIFFERENT
 7008  030544  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 12
 7009  030546  001016                        BNE     4$              ,BRANCH IF DIFFERENT
 7010  030550  162321                        SUB     (R3)+,(R1)+     ,CHECK WORD 13
 7011  030552  001014                        BNE     4$              ,BRANCH IF DIFFERENT
```

```
7012  030554  162321                    SUB     (R3)+,(R1)+     ;CHECK WORD 14
7013  030556  001012                    BNE     4$              ,BRANCH IF DIFFERENT
7014  030560  162321                    SUB     (R3)+,(R1)+     ;CHECK WORD 15
7015  030562  001010                    BNE     4$              ;BRANCH IF DIFFERENT
7016  030564  162321                    SUB     (R3)+,(R1)+     ;CHECK WORD 16
7017  030566  001006                    BNE     4$              ;BRANCH IF DIFFERENT
7018  030570  062702  000020            ADD     #16 ,R2         ;DONE ?
7019  030574  001333                    BNE     1$              ;NO--BRANCH
7020  030576  005726              3$    TST     (SP)+           ;YES -- CLEAN UP STACK
7021  030600  104413                    RESREG                  ;RESTORE R0 - R5
7022  030602  000207                    RTS     PC
7023  030604  010104              4$    MOV     R1,R4           ;FORM THE WORD NUMBER
7024  030606  162704  047712            SUB     #BUFFER,R4
7025  030612  006204                    ASR     R4              ,WORD NUMBER
7026  030614  010305                    MOV     R3,R5           ;FORM ADDRESS TO CONTINUE FROM
7027  030616  166005  003044            SUB     PAT PT(R0),R5
7028  030622  006305                    ASL     R5
7029  030624  062705  030470            ADD     #2$,R5          ;ADDRESS
7030  030630  064341                    ADD     -(R3),-(R1)     ;RECONSTRUCT THE BAD WORD
7031  030632  010137  001122            MOV     R1,@#$BDADR     ;SAVE THE ERROR INFORMATION
7032  030636  010337  001120            MOV     R3,@#$GDADR     ,
7033  030642  012137  001126            MOV     (R1)+,@#$BDDAT  ,
7034  030646  012337  001124            MOV     (R3)+,@#$GDDAT  ;
7035  030652  005716                    TST     (SP)            ,1ST DATA COMPARE ERROR?
7036  030654  001023                    BNE     6$              ;NO--BRANCH
7037  030656  013737  004176  001270    MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
7038  030664  113737  004175  001274    MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
7039  030672  113737  004174  001272    MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
7040  030700  016600  000016            MOV     16(SP),R0       ;GET TEST PC+4
7041  030704  012737  030714  001206    MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
7042  030712  104013                    ERROR   13              ;REPORT THE ERROR
7043  030714  016600  000014        5$  MOV     14(SP),R0       ;PATTERN NUMBER INDEX
7044  030720  105116                    COMB    (SP)            ,SET THE ERROR SWITCH
7045  030722  000404                    BR      7$
7046  030724                        6$
7047  030724  012737  030734  001206    MOV     #7$,$ESCAPE     ;;ESCAPE TO 7$ ON ERROR
7048  030732  104014                    ERROR   14              ;REPORT THE ERROR
7049  030734  032777  000002  150176  7$  BIT     #SW01,@SWR      ,STOP DATA COMPARE?
7050  030742  001315                    BNE     3$              ,YES--EXIT
7051  030744  123737  001364  001103    CMPB    @#ERR.CT,@#$ERFLG ;MAX. ERRORS?
7052  030752  101004                    BHI     8$              ;NO--BRANCH
7053  030754  013766  001252  000016    MOV     @#BYPASS,16(SP) ,YES--ERROR EXIT
7054  030762  000705                    BR      3$
7055  030764  000115                8$  JMP     (R5)            ;NO--CONTINUE AT NEXT WORD
7056
7057                                    ,THIS ROUTINE WILL FILL THE DATA BUFFER (256*22 WORDS) WITH
7058                                    ,A RANDOM PATTERN.  THE FIRST TWO WORDS OF EVERY 256 WILL
7059                                    ,BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
7060                                    ,NEXT 254 WORDS.
7061                                    ,NOTE  THIS ROUTINE DESTROYS R1 AND R2
7062                                    ,CALL
7063                                    ,       JSR     R0,@#FILRAN
7064                                    ,       RETURN
7065
7066  030766  012701  047712        FILRAN  MOV     #BUFFER,R1
7067  030772  013702  001630                MOV     PRMLMT+22,R2    ;MAXIMUM NUMBER OF SECTORS
```

```
7068  030776  004037  031206          1$:     JSR     R0,@#RANPAT
7069  031002  005302                          DEC     R2
7070  031004  100374                          BPL     1$
7071  031006  000200                          RTS     R0
7072
7073                                  ;THIS ROUTINE USES THE FIRST TWO WORDS OF THE
7074                                  ;READ BUFFER TO GENERATED A RANDOM PATTERN   THEN
7075                                  ;THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED
7076                                  ;NOTE  THIS ROUTINE DESTROYS R1-R4
7077                                  ;CALL
7078                                  ;       JSR     R0,@#RANCK
7079                                  ;       RETURN
7080
7081  031010  013746  023606          RANCK:  MOV     @#$HINUM,-(SP)  ;SAVE THE PRESENT RANDOM NUMBER
7082  031014  013746  023610                  MOV     @#$LONUM,-(SP)
7083  031020  012702  050712                  MOV     #BUFFER+512.,R2 ;READ BUFFER ADDRESS
7084  031024  012701  051712                  MOV     #BUFFER+1024.,R1 ;RANDOM PATTERN ADDRESS
7085  031030  010103                          MOV     R1,R3           ;COPY IT INTO R3 FOR LATER USE
7086  031032  011237  023610                  MOV     (R2),@#$LONUM   ;PRIME THE RANDOM NUMBER GENERATOR
7087  031036  016237  000002  023606          MOV     2(R2),@#$HINUM
7088  031044  004037  031206                  JSR     R0,@#RANPAT     ;GENERATE A RANDOM PATTERN
7089  031050  012637  023610                  MOV     (SP)+,@#$LONUM  ;RESTORE PRESENT RANDOM NUMBER
7090  031054  012637  023606                  MOV     (SP)+,@#$HINUM
7091  031060  005046                          CLR     -(SP)           ;NO ERRORS
7092  031062  162322                  1$:     SUB     (R3)+,(R2)+     ;ARE THESE TWO WORDS DIFFERENT?
7093  031064  001441                          BEQ     4$              ;NO--BRANCH
7094  031066  012737  031140  001206          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
7095  031074  064342                          ADD     -(R3),-(R2)     ;RECREATE THE BAD WORD
7096  031076  010237  001122                  MOV     R2,@#$BDADR     ;ADDRESS OF BAD DATA
7097  031102  010337  001120                  MOV     R3,@#$GDADR     ;ADDRESS OF GOOD DATA
7098  031106  012237  001126                  MOV     (R2)+,@#$BDDAT  ;BAD DATA
7099  031112  012337  001124                  MOV     (R3)+,@#$GDDAT  ;GOOD DATA
7100  031116  010204                          MOV     R2,R4           ;FORM WORD NUMBER (1 TO 256)
7101  031120  162704  050712                  SUB     #BUFFER+512 ,R4
7102  031124  006204                          ASR     R4
7103  031126  005716                          TST     (SP)            ;FIRST ERROR
7104  031130  001002                          BNE     2$              ;NO--BRANCH
7105  031132  105116                          COMB    (SP)            ;YES--SET ERROR SWITCH
7106  031134  104015                          ERROR   15              ;REPORT THE ERROR
7107  031136  104016                  2$:     ERROR   16              ;REPORT THE ERROR
7108  031140  032777  001000  147772  3$:     BIT     #SW09,@SWR      ;LOOP ON ERROR?
7109  031146  001012                          BNE     5$              ;YES--BRANCH
7110  031150  123737  001364  001103          CMPB    @#ERR.CT,@#$ERFLG ;MAX. ERRORS OCCURRED?
7111  031156  101406                          BLOS    5$              ;YES--BRANCH
7112  031160  032777  000002  147752          BIT     #SW01,@SWR      ;STOP COMPARING?
7113  031166  001002                          BNE     5$              ;YES--BRANCH
7114  031170  020103                  4$:     CMP     R1,R3           ;ALL DATA BEEN COMPARED?
7115  031172  101333                          BHI     1$              ;NO--BRANCH
7116  031174  005726                  5$:     TST     (SP)+           ;ERROR OCCUR?
7117  031176  001402                          BEQ     6$              ;NO--BRANCH
7118  031200  013700  001252                  MOV     @#BYPASS,R0     ;TAKE ERROR EXIT
7119  031204  000200                  6$:     RTS     R0              ;EXIT
7120
7121                                  ;THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
7122                                  ;PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
7123                                  ;OF THE PATTERN
```

```
7124                                    ;CALL
7125                                    ;       MOV     #ADR,R1         ;ADDRESS OF THE BUFFER
7126                                    ;       JSR     R0,@#RANPAT
7127                                    ;       RETURN
7128
7129  031206  010246                    RANPAT  MOV     R2,-(SP)        ;SAVE R2
7130  031210  012702  000200                    MOV     #256./2.,R2     ;GENERATE 256 WORDS
7131  031214  000402                            BR      2$
7132  031216  004737  023510            1$      JSR     PC,@#$RAND      ;GENERATE A RANDOM NUMBER
7133  031222  013721  023610            2$      MOV     @#$LONUM,(R1)+  ;PUT LOW WORD IN BUFFER
7134  031226  013721  023606                    MOV     @#$HINUM,(R1)+  ;PUT HIGH WORD IN BUFFER
7135  031232  005302                            DEC     R2              ;DONE?
7136  031234  003370                            BGT     1$              ;NO--BRANCH
7137  031236  012602                            MOV     (SP)+,R2        ;RESTORE R2
7138  031240  000200                            RTS     R0              ;EXIT
7139
7140                                    ;THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
7141                                    ;ADDRESSES AND SAVES THEN IN THE DPB (DTADPB+10 AND DTADPB+12)
7142                                    ;NOTE.  THIS ROUTINE DESTROYS R1-R3
7143                                    ;CALL
7144                                    ;       JSR     R0,@#RANADR
7145                                    ;       RETURN
7146
7147  031242  004737  023510            RANADR. JSR     PC,@#$RAND      ;GENERATE A RANDOM NUMBER
7148  031246  113701  023610                    MOVB    @#$LONUM,R1     ;FORM SECTOR IN R1
7149  031252  113702  023611                    MOVB    @#$LONUM+1,R2   ;FORM TRACK IN R2
7150  031256  013703  023606                    MOV     @#$HINUM,R3     ;FORM CYLINDER IN R3
7151  031262  105701                            TSTB    R1              ;ENSURE THE SECTOR IS BETWEEN 0 AND 21
7152  031264  002403                            BLT     2$
7153  031266  123701  001630            1$      CMPB    PRMLMT+22,R1    ;CHECK MAXIMUM SECTOR ADDRESS
7154  031272  103003                            BHIS    3$
7155  031274  000241            2$      CLC
7156  031276  106001                    RORB    R1
7157  031300  000772                    BR      1$
7158  031302  105702            3$      TSTB    R2              ;ENSURE THE TRACK IS BETWEEN 0 AND 18
7159  031304  002403                    BLT     5$
7160  031306  122702  000023            4$      CMPB    #19.,R2
7161  031312  003003                    BGT     6$
7162  031314  000241            5$      CLC
7163  031316  106002                    RORB    R2
7164  031320  000772                    BR      4$
7165  031322  023703  001510            6$      CMP     @#FC,R3         ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
7166  031326  003413                    BLE     7$
7167  031330  000241                    CLC
7168  031332  006003                    ROR     R3
7169  031334  005503                    ADC     R3
7170  031336  001371                    BNE     6$
7171  031340  010103                    MOV     R1,R3
7172  031342  000303                    SWAB    R3
7173  031344  060203                    ADD     R2,R3
7174  031346  005203                    INC     R3
7175  031350  003364                    BGT     6$
7176  031352  005403                    NEG     R3
7177  031354  000762                    BR      6$
7178  031356  023703  001512            7$      CMP     @#LC,R3
7179  031362  002003                    BGE     8$
```

```
7180  031364  000241                          CLC
7181  031366  006003                          ROR     R3
7182  031370  000772                          BR      7$
7183  031372  023703  001510          8$      CMP     @#FC,R3
7184  031376  003403                          BLE     9$
7185  031400  005203                          INC     R3
7186  031402  000303                          SWAB    R3
7187  031404  000764                          BR      7$
7188  031406  110137  004174          9$      MOVB    R1,@#DTADPB+10  ;SAVE SECTOR ADDRESS
7189  031412  110237  004175                  MOVB    R2,@#DTADPB+11  ,SAVE TRACK ADDRESS
7190  031416  010337  004176                  MOV     R3,@#DTADPB+12  ;SAVE CYLINDER ADDRESS
7191  031422  000200                          RTS     R0              ;RETURN
7192
7193                                  ,THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
7194                                  ,IF SWR<07>=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
7195                                  ;SETTING IS READ AND STORED.
7196                                  ,NOTE  THIS ROUTINE DESTROYS R3 AND R4
7197                                  ,CALL
7198                                  ,       JSR     PC,@#GETSWR
7199                                  ,       RETURN                  ;(C.SWR)=DESIRED CONTROL SWITCHES
7200
7201  031424  032777  000200  147506  GETSWR  BIT     #SW07,@SWR      ;READ CONTROL SWITCHES?
7202  031432  001430                          BEQ     2$              ,NO--BRANCH
7203  031434  104401  031442                  TYPE    ,65$            ,,TYPE ASCIZ STRING
7204  031440  000410                          BR      64$             ,;GET OVER THE ASCIZ
7205                                  ,,65$    ASCIZ  <CR><LF>/SET SWR<07>=0/
7206  031462                          64$.
7207  031462  012703  043050          1$      MOV     #MSG CS,R3      ;"CONTROL SWITCHES="
7208  031466  013704  001220                  MOV     @#C SWR,R4      ,PRESENT CONTROL SWITCH SETTINGS
7209  031472  004037  031516                  JSR     R0,@#GETNUM     ,GET THE NEW SWITCH SETTINGS
7210  031476  000771                          BR      1$              , COMMA
7211  031500  000240                          NOP                     ;PERIOD
7212  031502  013737  001220  001222          MOV     C SWR,SAVCSW    ,SAVE PREVIOUS VALUE
7213  031510  010437  001220                  MOV     R4,@#C SWR      , DOUBLE PERIOD-SAVE NEW SWITCH SETTING
7214  031514  000207                  2$      RTS     PC              ;RETURN FROM CALL
7215
7216                                  ,THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
7217                                  ,INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
7218                                  ,IF REQUIRED
7219                                  ;NOTE  THIS ROUTINE DESTROYS R1
7220                                  ,CALL
7221                                  ,       MOV     #ADR,R3         ,ADDRESS OF ASCIZ MESSAGE
7222                                  ,       MOV     #NUM,R4         ,OCTAL NUMBER
7223                                  ,       JSR     R0,@#GETNUM
7224                                  ,       RETURN1                 ,INPUT TERMINATED WITH A COMMA
7225                                  ,       RETURN2                 ,WITH A PERIOD
7226                                  ,       RETURN3                 ,WITH A DOUBLE PERIOD
7227                                  ,                               ,R4=INPUT NUMBER AND
7228                                  ,                               ,R2=R4*32 FOR ALL
7229                                  ,                               ,THREE RETURNS
7230
7231  031516  010337  031524          GETNUM  MOV     R3,2$           ;SAVE MESSAGE POINTER
7232  031522  104401                  1$      TYPE                    ,TYPE THE MESSAGE
7233  031524  000000                  2$.      WORD   0               ,MESSAGE POINTER GOES HERE
7234  031526  010446                          MOV     R4,-(SP)        ,,SAVE R4 FOR TYPEOUT
7235  031530  104402                          TYPOC                   ,,GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
7236  031532  104401  043074                TYPE    ,SLASH          ,' / '
7237  031536  104411                        RDLIN                   ;READ AN ASCIZ STRING
7238  031540  012601                        MOV     (SP)+,R1        ,ADDRESS OF FIRST CHARACTER
7239  031542  004037  033766                JSR     R0,@#CK CHR     ;CHECK ONE CHARACTER
7240  031546  031522                        1$                      ;ILLEGAL CHARACTER
7241  031550  031522                        1$                      ,CARRIAGE RETURN
7242  031552  031564                        3$                      ;"/"
7243  031554  031610                        7$                      ;","
7244  031556  031616                        8$                      ;"."
7245  031560  031562                        11$                     ;DIGIT 0-9
7246  031562  005301                11$     DEC     R1              ,DECREMENT THE INPUT POINTER
7247  031564                        3$
7248  031564  004037  034226                JSR     R0,@#CK NUM     ;CHECK THE NUMBER
7249  031570  031522                        1$                      ;ILLEGAL INPUT
7250  031572  031604                        6$                      ;TERMINATED WITH A "," OR "CR"
7251  031574  031602                        5$                      ;TERMINATED WITH A " "
7252  031576  031600                        4$                      ,TERMINATED WITH A " ."
7253  031600  005720                4$      TST     (R0)+           ,DOUBLE PERIOD
7254  031602  005720                5$      TST     (R0)+           ,SINGLE PERIOD
7255  031604  010204                6$      MOV     R2,R4           ,COMMA--SAVE INPUT NUMBER
7256  031606  000414                        BR      10$             ,GO TO EXIT
7257  031610  105711                7$      TSTB    (R1)            ,TERMINATOR AFTER A COMMA?
7258  031612  001343                        BNE     1$              ,NO--LOOP
7259  031614  000411                        BR      10$             ,YES--EXIT
7260  031616  105711                8$      TSTB    (R1)            ,TERMINATOR AFTER A PERIOD?
7261  031620  001406                        BEQ     9$              ,YES--EXIT
7262  031622  122721  000056                CMPB    #'.,(R1)+       ,NO--DOUBLE PERIOD?
7263  031626  001335                        BNE     1$              ,NO--LOOP
7264  031630  105711                        TSTB    (R1)            ,YES--TERMINATOR?
7265  031632  001333                        BNE     1$              ,NO--LOOP
7266  031634  005720                        TST     (R0)+           ,DOUBLE PERIOD
7267  031636  005720                9$      TST     (R0)+           ,PERIOD
7268  031640  010402                10$     MOV     R4,R2           ,COMMA--POSITION THE
7269  031642  000302                        SWAB    R2              ,NUMBER IN CASE IT
7270  031644  006202                        ASR     R2              ,IS THE PRIORITY LEVEL
7271  031646  006202                        ASR     R2
7272  031650  006202                        ASR     R2
7273  031652  000200                        RTS     R0              ,EXIT
7274
7275                                ,THIS ROUTINE IS USED TO CHANGE OR MODIFY
7276                                ,THE TEST PARAMETERS   IT GIVES THE OPERATOR
7277                                ,THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
7278                                ,TESTS TO RUN AND HOW MANY TIMES TO
7279                                ,REPEAT EACH TEST
7280
7281  031654  104412                GT PRM  SAVREG                  ,SAVE R0 - R5
7282  031656  005037  001232        GT PR1  CLR     DRVSEL          ,NO DRIVE SELECTED
7283  031662  104401  031670                TYPE    ,65$            ,,TYPE ASCIZ STRING
7284  031666  000406                        BR      64$             ,,GET OVER THE ASCIZ
7285                                ,,65$   .ASCIZ  <CR><LF>/DRIVE(S)=/
7286  031704                        64$
7287  031704  104411                        RDLIN                   ,READ TTY
7288  031706  012601                        MOV     (SP)+,R1        ,ADDRESS OF ASCIZ STRING
7289  031710  004037  033766                JSR     R0,@#CK CHR     ,CHECK ONE CHARACTER
7290  031714  031656                        GT PR1                  ,ILLEGAL CHARACTER
7291  031716  031656                        GT PR1                  ,CARRIAGE RETURN
```

```
 7292  031720  031656                        GT PR1                  ,"/"
 7293  031722  031656                        GT PR1                  ;","
 7294  031724  031656                        GT PR1                  ," "
 7295  031726  031730                        1$                      ,DIGIT 0-9
 7296  031730  005301                  1$    DEC     R1
 7297  031732                        2$
 7298  031732  012702  000007              MOV     #7,R2   ,UPPER LIMIT OF INPUT
 7299  031736  004037  034042              JSR     R0,@#CK DIG     ,CHECK THE DIGIT(S)
 7300  031742  031656                        GT PR1                  ,ILLEGAL INPUT
 7301  031744  031656                        GT PR1                  ,INPUT TO LARGE
 7302  031746  031754                        3$                      ;TERMINATED WITH A "," OR "CR"
 7303  031750  032000                        4$                      ;TERMINATED WITH A " "
 7304  031752  032000                        4$                      ,TERMINATED WITH A "  "
 7305  031754  156237  034502  001232  3$    BISB    ATABIT(R2),DRVSEL ,SET THE DRIVE SELECTED BIT
 7306  031762  105741                        TSTB    -(R1)           ,WAS THE LINE TERMINATED?
 7307  031764  001362                        BNE     2$              ;NO-GET THE NEXT DRIVE
 7308  031766  005037  001234              CLR     @#TSTNMS        ,DESELECT ALL TESTS
 7309  031772  005037  001236              CLR     TSTNMS+2
 7310  031776  000405                        BR      GTTST1          ,YES--SELECT TEST
 7311  032000  156237  034502  001232  4$    BISB    ATABIT(R2),DRVSEL ,SET THE SELECTED DRIVE BITS
 7312  032006  104413                  GT PR2  RESREG                  ,RESTORE R0 - R5
 7313  032010  000207                        RTS     PC              ,EXIT
 7314
 7315  032012                        GTTST1
 7316  032012  104401  032020              TYPE    ,65$            ,,TYPE ASCIZ STRING
 7317  032016  000403                        BR      64$             ,,GET OVER THE ASCIZ
 7318                                ,,65$    ASCIZ   /TEST=/
 7319  032026                        64$
 7320  032026  104411                        RDLIN                   ,READ AN ASCIZ STRING
 7321  032030  012601                        MOV     (SP)+,R1        ,POINTER TO R1
 7322  032032  122711  000056              CMPB    #' ,(R1)        ,DOUBLE PERIOD?
 7323  032036  001007                        BNE     1$              ,NO--BRANCH
 7324  032040  122761  000056  000001        CMPB    #' ,1(R1)
 7325  032046  001003                        BNE     1$
 7326  032050  105761  000002              TSTB    2(R1)           ;"CR"?
 7327  032054  001754                        BEQ     GT PR2          ,YES--EXIT
 7328  032056  005037  001234          1$    CLR     TSTNMS          ,NO TEST SELECTED
 7329  032062  005037  001236              CLR     TSTNMS+2
 7330  032066  005037  001240              CLR     OPNFLG          ,NO TESTS TO BE OPENED
 7331  032072  005037  001242              CLR     OPNFLG+2
 7332  032076  121127  000123        GTTST2  CMPB    (R1),#'S        ,ALL SEEK TESTS?
 7333  032102  001004                        BNE     1$              ,NO--BRANCH
 7334  032104  052737  000777  001234        BIS     #777,TSTNMS     ;YES--SELECT TESTS 0-10
 7335  032112  000552                        BR      GTTST3
 7336  032114  121127  000124          1$    CMPB    (R1),#'T        ,ALL TIMING TESTS?
 7337  032120  001004                        BNE     2$              ;NO--BRANCH
 7338  032122  052737  036000  001234        BIS     #36000,TSTNMS   ,YES--SELECT TESTS 12-15
 7339  032130  000543                        BR      GTTST3
 7340  032132  121127  000101          2$    CMPB    (R1),#'A        ;ALL ADDRESSING TESTS?
 7341  032136  001004                        BNE     3$              ;NO--BRANCH
 7342  032140  052737  140000  001234        BIS     #140000,TSTNMS  ,YES--SELECT TESTS 16 & 17
 7343  032146  000534                        BR      GTTST3
 7344  032150  121127  000104          3$    CMPB    (R1),#'D        ,DATA TEST?
 7345  032154  001004                        BNE     4$              ,NO--BRANCH
 7346  032156  052737  000001  001236        BIS     #1,TSTNMS+2     ,YES--SELECT TEST 20
 7347  032164  000525                        BR      GTTST3
```

```
 7348  032166  121127  000105          4$    CMPB    (R1),#'E          ,EXERCISER TEST?
 7349  032172  001004                        BNE     5$                ,NO--BRANCH
 7350  032174  052737  000002  001236        BIS     #2,TSTNMS+2       ;YES--SELECT TEST 21
 7351  032202  000516                        BR      GTTST3
 7352  032204  004037  033712          5$    JSR     R0,@#CK OCT       ;OCTAL DIGIT?
 7353  032210  000514                        BR      GTTST4            ,NO--BRANCH
 7354  032212  010205                        MOV     R2,R5             ;YES--SAVE IT
 7355  032214  005201                        INC     R1                ;MOVE TO NEXT CHARACTER
 7356  032216  004037  033712                JSR     R0,@#CK OCT       ;OCTAL DIGIT
 7357  032222  000405                        BR      6$                ;NO--BRANCH
 7358  032224  005201                        INC     R1                ,MOVE TO NEXT CHARACTER
 7359  032226  006305                        ASL     R5                ,SCALE HIGH DIGIT
 7360  032230  006305                        ASL     R5
 7361  032232  006305                        ASL     R5
 7362  032234  060502                        ADD     R5,R2             ,COMBINE HIGH & LOW DIGITS
 7363  032236  020227  000022          6$    CMP     R2,#$TN-1         ;VALID TEST NUMBER?
 7364  032242  003263                        BGT     GTTST1            ,NO--BRANCH
 7365  032244  010237  032436                MOV     R2,9$             ,SAVE THE TEST NUMBER
 7366  032250  010204                        MOV     R2,R4             ,CONVERT TEST NUMBER INTO AN INDEX
 7367  032252  042704  000017                BIC     #17,R4            ,CLEAR UNWANTED BITS
 7368  032256  006204                        ASR     R4                ,SHIFT THE BITS
 7369  032260  006204                        ASR     R4                ,SHIFT THE BITS
 7370  032262  006204                        ASR     R4                ,SHIFT THE BITS
 7371  032264  006302                        ASL     R2
 7372  032266  056264  001424  001234        BIS     BITS(R2),TSTNMS(R4)   ,SELECT TEST
 7373  032274  121127  000055                CMPB    (R1),#'-          ,TEST STRING?
 7374  032300  001060                        BNE     GTTST4            ,NO--BRANCH
 7375  032302  005201                        INC     R1                ,YES--MOVE TO NEXT CHARACTER
 7376  032304  004037  033712                JSR     R0,@#CK.OCT       ,OCTAL DIGIT?
 7377  032310  000640                        BR      GTTST1            ,NO--BRANCH
 7378  032312  010205                        MOV     R2,R5             ,YES--SAVE IT
 7379  032314  005201                        INC     R1                ,MOVE TO NEXT CHARACTER
 7380  032316  004037  033712                JSR     R0,@#CK OCT       ,OCTAL DIGIT?
 7381  032322  000405                        BR      7$                ,NO--BRANCH
 7382  032324  005201                        INC     R1                ,YES--MOVE TO NEXT CHARACTER
 7383  032326  006305                        ASL     R5                ,SCALE HIGH DIGIT
 7384  032330  006305                        ASL     R5
 7385  032332  006305                        ASL     R5
 7386  032334  060502                        ADD     R5,R2             ,COMBINE HIGH & LOW DIGIT
 7387  032336  020227  000022          7$    CMP     R2,#$TN-1         ,VALID TEST NUMBER?
 7388  032342  003223                        BGT     GTTST1            ,NO--BRANCH
 7389  032344  023702  032436                CMP     9$,R2             ,IS THE FIRST NUMBER OF THE
 7390                                                                  ,STRING SMALLER THAN THE LAST?
 7391  032350  002220                        BGE     GTTST1            ,NO--BRANCH
 7392  032352  010246                        MOV     R2,-(SP)          ,SAVE ENDING TEST NUMBER
 7393  032354  013702  032436                MOV     9$,R2             ,GET STARTING TEST NUMBER
 7394  032360  012637  032436                MOV     (SP)+,9$          ,STORE ENDING TEST NUMBER
 7395  032364  006337  032436                ASL     9$                ,SHIFT ENDING TEST NUMBER
 7396  032370  006302                        ASL     R2                ,SHIFT TEST NUMBER
 7397  032372  010204          8$    MOV     R2,R4             ,COPY TEST NUMBER INTO R4
 7398  032374  042704  000037                BIC     #37,R4            ,CLEAR LOWER BITS
 7399  032400  006204                        ASR     R4                ,SHIFT THE TEST NUMBER
 7400  032402  006204                        ASR     R4                ,SHIFT THE TEST NUMBER
 7401  032404  006204                        ASR     R4                ,SHIFT THE TEST NUMBER
 7402  032406  006204                        ASR     R4                ,SHIFT THE TEST NUMBER
 7403  032410  056264  001424  001234        BIS     BITS(R2),TSTNMS(R4)   ,SELECT THE TEST
```

```
7404  032416  062702  000002              ADD     #2,R2           ;INECREMENT THE TEST NUMBER
7405  032422  020237  032436              CMP     R2,9$           ;SEE IF FINISHED
7406  032426  101761                      BLOS    8$              ;BR IF NOT
7407  032430  162702  000002              SUB     #2,R2           ;CORRECT TEST NUMBER
7408  032434  000402                      BR      GTTST4          ;CONTINUE
7409  032436  000000              9$       WORD    0               ;STORE TEST NUMBER HERE
7410  032440  005201              GTTST3  INC     R1              ;MOVE TO NEXT CHARACTER
7411  032442  121127  000056      GTTST4  CMPB    (R1),#'         ;"PERIOD"?
7412  032446  001441                      BEQ     GTTST5          ;YES--BRANCH
7413  032450  005737  001234              TST     TSTNMS          ;ANY TEST SELECTED THIS CYCLE?
7414  032454  0C1005                      BNE     1$              ;BR IF YES
7415  032456  005737  001236              TST     TSTNMS+2        ;ANY TEST SELECTED THIS CYCLE ?
7416  032462  001002                      BNE     1$              ;BR IF YES
7417  032464  000137  032012              JMP     GTTST1          ;NO
7418  032470  121127  000057      1$      CMPB    (R1),#'/        ;"OPEN"?
7419  032474  001004                      BNE     2$              ;NO--BRANCH
7420  032476  056264  001424  001240      BIS     BITS(R2), OPNFLG(R4)  ;YES--SET BITS FOR TEST TO OPEN
7421  032504  000405                      BR      3$
7422  032506  121127  000054      2$      CMPB    (P1),#',        ;"COMMA"?
7423  032512  001402                      BEQ     3$              ;BR IF YES
7424  032514  000137  032012              JMP     GTTSTi          ;NO
7425  032520  005201              3$      INC     R1              ;MOVE TO NEXT CHARACTER
7426  032522  105711                      TSTB    (R1)            ;"CR"?
7427  032524  001402                      BEQ     4$              ;BR IF 'CR'
7428  032526  000137  032076              JMP     GTTST2          ;NO--GO GET NEXT CHARACTER
7429  032532  005737  001240      4$      TST     OPNFLG          ;ANY TESTS TO OPEN ?
7430  032536  001C42                      BNE     OPNTST          ;BR IF YES
7431  032540  005737  001242              TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
7432  032544  001037                      BNE     OPNTST          ;BR IF YES
7433  032546  000137  032012              JMP     GTTST1          ;NO--START AGAIN
7434  032552  005201              GTTST5  INC     R1              ;MOVE TO NEXT CHARACTER
7435  032554  121127  000056              CMPB    (R1),#'         ;"PERIOD"?
7436  032560  001414                      BEQ     GTTST6          ;YES--BRANCH
7437  032562  105711                      TSTB    (R1)            ;"CR"?
7438  032564  001402                      BEQ     1$              ;YES--BRANCH
7439  032566  000137  032012              JMP     GTTST1
7440  032572  005737  001240      1$      TST     OPNFLG          ;ANY TESTS TO OPEN ?
7441  032576  001022                      BNE     OPNTST          ;BR IF YES
7442  032600  005737  001242              TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
7443  032604  001017                      BNE     OPNTST          ;BR IF YES
7444  032606  000137  032006              JMP     GT PR2          ;NO--GO START TESTING
7445  032612  005201              GTTST6  INC     R1              ;MOVE TO NEXT CHARACTER
7446  032614  105711                      TSTB    (R1)            ;"CR"?
7447  032616  001402                      BEQ     1$              ;YES--BRANCH
7448  032620  000137  032012              JMP     GTTST1          ;NO--GO ASK FOR TEST
7449  032624  005737  001240      1$      TST     OPNFLG          ;ANY TESTS TO OPEN ?
7450  032630  001005                      BNE     OPNTST          ;BR IF YES
7451  032632  005737  001242              TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
7452  032636  001002                      BNE     OPNTST          ;BR IF YES
7453  032640  000137  032006              JMP     GT.PR2          ;NO--GO START TESTING
7454
7455                                      ;OPEN THE SELECTED TEST FOR CHANGES
7456
7457  032644  104412              OPNTST  SAVREG                  ;SAVE R0 - R5
7458  032646  005027                      CLR     (PC)+           ;START WITH TEST 0
7459  032650  000000              OPN CT   WORD   0               ;COUNT STORED HERE
```

```
7460  032652  000411                          BR      OPN 2           ;SKIP THE INCREMENT
7461  032654  005237  032650          OPN 1   INC     OPN CT          ;MOVE TO THE NEXT TEST
7462  032660  022737  000022  032650          CMP     #$TN-1,OPN CT   ,TEST NUMBER TOO BIG?
7463  032666  002003                          BGE     OPN 2           ;NO--OPEN THE NEXT TEST
7464  032670  104413                          RESREG                  ;RESTORE R0 - R5
7465  032672  000137  032012                  JMP     GTTST1          ;YES--GO ASK FOR MORE TESTS
7466  032676  013705  032650          OPN 2   MOV     OPN.CT,R5       ;SETUP TO USE THE
7467  032702  006305                          ASL     R5              ;TEST NUMBER AS AN INDEX
7468  032704  013703  032650                  MOV     OPN CT,R3       ,GET INDEX
7469  032710  042703  000017                  BIC     #17,R3          ;CLEAR LOWER TEST BITS
7470  032714  006203                          ASR     R3              ;SHIFT TEST NUMBER
7471  032716  006203                          ASR     R3              ,SHIFT TEST NUMBER
7472  032720  006203                          ASR     R3              ,SHIFT TEST NUMBER
7473  032722  036563  001424  001240          BIT     BITS(R5),OPNFLG(R3)  ;OPEN THIS TEST?
7474  032730  001751                          BEQ     OPN 1           ,NO--MOVE TO NEXT TEST
7475  032732  104401  032740                  TYPE    ,65$            ,,TYPE ASCIZ STRING
7476  032736  000404                          BR      64$             ,,GET OVER THE ASCIZ
7477                          ,,65$    ASCIZ  / TEST /
7478  032750                  64$
7479  032750  013746  032650                  MOV     OPN CT,-(SP)    ,,SAVE OPN.CT FOR TYPEOUT
7480                                                                  ,,TEST NUMBER
7481  032754  104403                          TYPOS                   ,,GO TYPE--OCTAL ASCII
7482  032756     002                           BYTE    2               ,,TYPE 2 DIGIT(S)
7483  032757     000                           BYTE    0               ,,SUPPRESS LEADING ZEROS
7484  032760  104401  001215                  TYPE    ,$CRLF          ,TYPE "CR" & "LF"
7485  032764  016500  001536                  MOV     PRMPT(R5),R0    ,PICKUP PARAMETER POINTER
7486  032770  011046                          MOV     (R0),-(SP)      ,SAVE THE VARIABLE INDICATOR
7487  032772  012702  001504                  MOV     #PRM,R2         ,FIRST ADDRESS OF TABLE
7488  032776  000405                          BR      2$
7489  033000  006216                  1$      ASR     (SP)            ,CHECK FOR A VARIABLE
7490  033002  103403                          BCS     2$              ,GO MOVE THIS ONE
7491  033004  001404                          BEQ     OPNPRM          ;DONE
7492  033006  005722                          TST     (R2)+           ,BUMP THE POINTER
7493  033010  000773                          BR      1$
7494  033012  012022                  2$      MOV     (R0)+,(R2)+     ,MOVE THIS VARIABLE INTO THE
7495  033014  000771                          BR      1$              ,COMMON AREA
7496  033016  013716  001504          OPNPRM  MOV     @#PRM,(SP)      ,GET THE VARIABLE INDICATOR
7497  033022  005004                          CLR     R4              ,ZERO THE INDEX
7498  033024  006216                  1$      ASR     (SP)            ,CHECK FOR A VARIABLE
7499  033026  103403                          BCS     3$              ,GO GET IT
7500  033030  001772                          BEQ     OPNPRM          ;OUT OF VARIABLES
7501  033032  005724                  2$      TST     (R4)+           ,UPDATE THE INDEX
7502  033034  000773                          BR      1$
7503  033036  005764  001606          3$      TST     PRMLMT(R4)      ,IS THE MAX  MAGNITUDE NEG?
7504  033042  100466                          BMI     OPNPAT          ,YES--THEN IT IS THE PATTERN
7505  033044  104401  044040                  TYPE    ,MSG SP         ,TYPE SPACES
7506  033050  016437  001636  033060          MOV     PRMMSG(R4),4$   ,TYPE THE NAME OF THIS VARIABLE
7507  033056  104401                          TYPE
7508  033060  000000                  4$       WORD   0
7509  033062  104401  043046                  TYPE    ,MSG EQ         ,TYPE "='
7510  033066  016446  001506                  MOV     RPT(R4),-(SP)           ,PUT RPT(R4) ON THE STACK
7511  033072  004737  023220                  JSR     PC,@#$SB2D      ,CHANGE RPT(R4) TO DECIMAL ASCIZ
7512  033076  004737  023450                  JSR     PC,@#$SUPRS     ,TYPE WITHOUT LEADING ZEROS
7513  033102  104401  043074                  TYPE    ,SLASH          ,' / '
7514  033106  104411                          RDLIN
7515  033110  012601                          MOV     (SP)+,R1        ,READ AN ASCIZ STRING
```

```
7516  033112  004037  033766          JSR     R0,@#CK CHR     ;CHECK ONE CHARACTER
7517  033116  033036                  3$                      ;ILLEGAL CHARACTER
7518  033120  033032                  2$                      ;CARRIAGE RETURN
7519  033122  033170                  8$                      ;"/"
7520  033124  033132                  5$                      ;","
7521  033126  033140                  6$                      ;"."
7522  033130  033166                  7$                      ;DIGIT 0-9
7523  033132  105711          5$      TSTB    (R1)            ,"CR"?
7524  033134  001340                  BNE     3$              ;NO--STAY ON THIS VARIABLE
7525  033136  000735                  BR      2$              ;YES--MOVE TO NEXT VARIABLE
7526  033140  105711          6$      TSTB    (R1)            ;IS THERE A "CR" AFTER THE PERIOD?
7527  033142  001002                  BNE     64$             ,NO
7528  033144  000137  033560          JMP     OPN N2          ,YES--GO CLOSE THIS TEST
7529  033150  122721  000056  64$     CMPB    #'.,(R1)+       ,DOUBLE PERIOD?
7530  033154  001330                  BNE     3$              ,NO--GO ASK FOR THIS VARIABLE
7531  033156  105711                  TSTB    (R1)            ,YES--IS A "CR" AFTER THE DOUBLE PERIOD?
7532  033160  001326                  BNE     3$              ,NO--ASK FOR THIS VARIABLE AGAIN
7533  033162  000137  033576          JMP     OPN X2          ,YES--CLOSE ALL TEST
7534  033166  005301          7$      DEC     R1              ,BACK THE POINTER UP BY ONE
7535  033170                  8$
7536  033170  016402  001606          MOV     PRMLMT(R4),R2   ,UPPER LIMIT OF INPUT
7537  033174  004037  034042          JSR     R0,@#CK DIG     ,CHECK THE DIGIT(S)
7538  033200  033036                  3$                      ,ILLEGAL INPUT
7539  033202  033036                  3$                      ,INPUT TO LARGE
7540  033204  033212                  9$                      ,TERMINATED WITH A "," OR "CR"
7541  033206  033554                  OPN N1                  ;TERMINATED WITH A " "
7542  033210  033572                  OPN X1                  ;TERMINATED WITH A "  "
7543  033212  010264  001506  9$      MOV     R2,RPT(R4)      ,SAVE THIS VARIABLE
7544  033216  000705                  BR      2$              ,MOVE TO NEXT VARIABLE
7545  033220  104401  044040  OPNPAT  TYPE    ,MSG SP         ,TYPE SPACES
7546  033224  104401  043042          TYPE    ,MSG PAT        ,TYPE "PAT"
7547  033230  104401  043046          TYPE    ,MSG EQ         ,TYPE "="
7548  033234  016446  001506          MOV     RPT(R4),-(SP)   ,,SAVE RPT(R4) FOR TYPEOUT
7549  033240  104402                  TYPOC                   ,,GO TYPE--OCTAL ASCII(ALL DIGITS)
7550  033242  104401  044041          TYPE    ,MSG SP+1       ,TYPE ONE SPACE
7551  033246  104411                  RDLIN                   ,READ ASCIZ STRING
7552  033250  012601                  MOV     (SP)+,R1        ,PICKUP POINTER
7553  033252  004037  033766          JSR     R0,@#CK CHR     ,CHECK ONE CHARACTER
7554  033256  033220                  OPNPAT                  ,ILLEGAL CHARACTER
7555  033260  033016                  OPNPRM                  ,CARRIAGE RETURN
7556  033262  033314                  3$                      ,"/"
7557  033264  033016                  OPNPRM                  ,","
7558  033266  033272                  1$                      ,"."
7559  033270  033312                  2$                      ,DIGIT 0-9
7560  033272  105711          1$      TSTB    (R1)            ,"CR" AFTER THE PERIOD?
7561  033274  001531                  BEQ     OPN N2          ,YES--GO CLOSE THIS TEST
7562  033276  122721  000056          CMPB    #'.,(R1)+       ,NO--PERIOD?
7563  033302  001346                  BNE     OPNPAT          ;NO--LOOP
7564  033304  105711                  TSTB    (R1)            ,"CR" AFTER A DOUBLE PERIOD?
7565  033306  001533                  BEQ     OPN X2          ,YES--GO START TESTING
7566  033310  000743                  BR      OPNPAT          ,NO--LOOP
7567  033312  005301          2$      DEC     R1              ,BACKUP THE ASCII POINTER
7568  033314                  3$
7569  033314  004037  034226          JSR     R0,@#CK NUM     ,CHECK THE NUMBER
7570  033320  033220                  OPNPAT                  ,ILLEGAL INPUT
7571  033322  033330                  4$                      ,TERMINATED WITH A "," OR "CR"
```

```
7572  033324  033554                        OPN.N1                  ;TERMINATED WITH A " "
7573  033326  033572                        OPN X1                  ;TERMINATED WITH A "  "
7574  033330  010264  001506          4$.   MOV     R2,RPT(R4)        ;SAVE THE INPUT NUMBER
7575  033334  006002                        ROR     R2                ,OPEN PATTERN 0?
7576  033336  103227                        BCC     OPNPRM            ;NO--START AT BEGINNING OF PARAMETER TABLE
7577  033340  104412                        SAVREG                    ;SAVE R0 - R5
7578  033342  005000          OPNWDS.       CLR     R0                ;START WITH WORD 0
7579  033344  012704  003104                MOV     #PAT0,R4
7580  033350                        1$.
7581  033350  104401  033356                TYPE    ,65$              ,,TYPE ASCIZ STRING
7582  033354  000403                        BR      64$               ,;GET OVER THE ASCIZ
7583                                ,,65$   .ASCIZ  /   WD/
7584  033364                        64$
7585  033364  010046                        MOV     R0,-(SP)                  ;PUT R0 ON THE STACK
7586  033366  004737  023220                JSR     PC,@#$SB2D        ;CHANGE R0 TO DECIMAL ASCIZ
7587  033372  004737  023450                JSR     PC,@#$SUPRS       ,TYPE WITHOUT LEADING ZEROS
7588  033376  104401  043046                TYPE    ,MSG EQ           ,TYPE "="
7589  033402  011446                        MOV     (R4),-(SP)        ,,SAVE (R4) FOR TYPEOUT
7590  033404  104402                        TYPOC                     ,;GO TYPE--OCTAL ASCII(ALL DIGITS)
7591  033406  104411                        RDLIN                     ;READ ASCIZ STRING
7592  033410  012601                        MOV     (SP)+,R1          ;PICKUP THE POINTER
7593  033412  004037  033766                JSR     R0,@#CK CHR       ,CHECK ONE CHARACTER
7594  033416  033350                        1$                        ;ILLEGAL CHARACTER
7595  033420  033452                        4$                        ,CARRIAGE RETURN
7596  033422  033434                        2$                        ,"/"
7597  033424  033452                        4$                        ,","
7598  033426  033466                        5$                        ;" "
7599  033430  033432                        10$                       ,DIGIT 0-9
7600  033432  005301          10$           DEC     R1                ,BACKUP THE ASCII POINTER
7601  033434                        2$
7602  033434  004037  034226                JSR     R0,@#CK NUM       ,CHECK THE NUMBER
7603  033440  033350                        1$                        ,ILLEGAL INPUT
7604  033442  033450                        3$                        ,TERMINATED WITH A "," OR "CR"
7605  033444  033506                        6$                        ,TERMINATED WITH A " "
7606  033446  033520                        8$                        ;TERMINATED WITH A "  "
7607  033450  010214                3$      MOV     R2,(R4)           ,SAVE THE INPUT
7608  033452  005724                4$      TST     (R4)+             ,MOVE TO NEXT WORD
7609  033454  005200                        INC     R0                ;INCREMENT THE COUNT
7610  033456  022700  000020                CMP     #16.,R0           ,COUNT TO LARGE?
7611  033462  003332                        BGT     1$                ;NO--BRANCH
7612  033464  000726                        BR      OPNWDS            ,YES--BRANCH
7613  033466  105711                5$      TSTB    (R1)              ,"CR" AFTER THE PERIOD?
7614  033470  001407                        BEQ     7$                ;YES--GO CLOSE THIS TEST
7615  033472  122721  000056                CMPB    #'.,(R1)+         ,NO--PERIOD?
7616  033476  001324                        BNE     1$                ,NO--BRANCH ILLEGAL INPUT STRING
7617  033500  105711                        TSTB    (R1)              ;"CR" AFTER THE "PERIOD-PERIOD"?
7618  033502  001407                        BEQ     9$                ,YES--GO START TESTING
7619  033504  000721                        BR      1$                ,NO--LOOP
7620  033506  010224                6$      MOV     R2,(R4)+          ;SAVE THE INPUT
7621  033510  004737  033532          7$    JSR     PC,@#CLSWDS       ;CLOSE THE DATA PATTERN
7622  033514  104413                        RESREG                    ,RESTORE R0 - R5
7623  033516  000420                        BR      OPN N2            ,MOVE TO NEXT TEST
7624  033520  010224                8$      MOV     R2,(R4)+          ,SAVE THE INPUT
7625  033522  004737  033532          9$    JSR     PC,@#CLSWDS       ,CLOSE THE DATA PATTERN
7626  033526  104413                        RESREG                    ,RESTORE R0 - R5
7627  033530  000422                        BR      OPN X2            ,START TESTING
```

```
 7628  033532  012701  003104          CLSWDS: MOV     #PAT0,R1        ;FIRST ADDRESS OF DATA PATTERN
 7629  033536  005200                  1$      INC     R0              ;COUNT THE LAST WORD THAT WAS STORED
 7630  033540  022700  000017                  CMP     #15.,R0         ;END OF TABLE
 7631  033544  002402                          BLT     2$              ,YES--EXIT
 7632  033546  012124                          MOV     (R1)+,(R4)+     ;COPY
 7633  033550  000772                          BR      1$              ;LOOP
 7634  033552  000207                  2$      RTS     PC              ;RETURN
 7635  033554  010264  001506          OPN N1  MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
 7636  033560  005726                  OPN N2  TST     (SP)+           ;CLEAN OFF THE STACK
 7637  033562  004737  033632                  JSR     PC,CLOSE        ;CLOSE THIS TEST
 7638  033566  000137  032654                  JMP     OPN 1           ;GO OPEN THE NEXT TEST
 7639  033572  010264  001506          OPN X1  MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
 7640  033576  005726                  OPN X2  TST     (SP)+           ;CLEAN OFF THE STACK
 7641  033600  004737  033632          1$      JSR     PC,CLOSE        ;CLOSE THIS TEST
 7642  033604  005725                  2$      TST     (R5)+           ;UPDATE THE INDEX
 7643  033606  020527  000034                  CMP     R5,#16*2        ;INDEX TO BIG?
 7644  033612  002403                          BLT     3$              ;NO--BRANCH
 7645  033614  104413                          RESREG                  ,RESTORE R0 - R5
 7646  033616  000137  032006                  JMP     GT PR2          ;GO TO EXIT
 7647  033622  036503  001424          3$      BIT     BITS(R5),R3     ;IS THIS TEST OPEN FOR CHANGE?
 7648  033626  001364                          BNE     1$              ,YES--GO CLOSE IT
 7649  033630  000765                          BR      2$              ;NO--MOVE TO NEXT TEST
 7650  033632  104412                  CLOSE   SAVREG                  ,SAVE R0 - R5
 7651  033634  012700  001504                  MOV     #PRM,R0         ,"FROM" ADDRESS
 7652  033640  016501  001536                  MOV     PRMPT(R5),R1    ;"TO" ADDRESS
 7653  033644  012002                          MOV     (R0)+,R2        ,"FROM" INDICATOR
 7654  033646  012103                          MOV     (R1)+,R3        ,"TO" INDICATOR
 7655  033650  012704  000001                  MOV     #1,R4           ,TEST BIT START A "RPT"
 7656  033654  030402                  1$      BIT     R4,R2           ,PARAMETER TO BE MOVED?
 7657  033656  001403                          BEQ     2$              ,NO--BRANCH
 7658  033660  030403                          BIT     R4,R3           ,A PLACE TO PUT IT?
 7659  033662  001404                          BEQ     3$              ,NO--BRANCH
 7660  033664  011011                          MOV     (R0),(R1)       ,YES--MOVE "FROM" TO "TO"
 7661  033666  030403                  2$      BIT     R4,R3           ,"TO" PARAMETER?
 7662  033670  001401                          BEQ     3$              ,NO--BRANCH
 7663  033672  005721                          TST     (R1)+           ,YES--UPDATE THE POINTER
 7664  033674  005720                  3$      TST     (R0)+           ,UPDATE FROM POINTER
 7665  033676  006304                          ASL     R4              ,POSITION THE TEST BIT
 7666  033700  032704  002000                  BIT     #BIT10,R4       ,DONE?
 7667  033704  001763                          BEQ     1$              ;NO--BRANCH
 7668  033706  104413                          RESREG                  ;RESTORE R0 - R5
 7669  033710  000207                          RTS     PC              ,RETURN
 7670                                  ,THIS ROUTINE IS USED TO CHECK IF AN
 7671                                  ,ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7
 7672                                  ,CALL
 7673                                  ,       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
 7674                                  ,       JSR     R0,@#CK OCT     ;CHECK THE CHARACTER
 7675                                  ,       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
 7676                                  ,       RETURN2                 ;CHARACTER IS IN R2 AS A
 7677                                  ,                               ;OCTAL DIGIT
 7678
 7679  033712  121127  000060          CK OCT  CMPB    (R1),#'0        ;LESS THAN ZERO?
 7680  033716  103407                          BLO     1$              ,YES -- BRANCH
 7681  033720  121127  000067                  CMPB    (R1),#'7        ,GREATER THAN SEVEN?
 7682  033724  101004                          BHI     1$              ,YES -- BRANCH
 7683  033726  111102                          MOVB    (R1),R2         ,GET THE CHARACTER
```

```
7684  033730  042702  177770          BIC     # C7,R2         ;STRIP AWAY THE ASCII
7685  033734  005720                  TST     (R0)+           ;ADJUST FOR RETURN
7686  033736  000200          1$      RTS     R0              ;RETURN
7687
7688                          ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
7689                          ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
7690                          ;CALL
7691                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
7692                          ;       JSR     R0,@#CK DEC     ;CHECK THE CHARACTER
7693                          ;       RETURN1                 ;NOT BETWEEN 0 AND 9
7694                          ;       RETURN2                 ;BETWEEN 0 AND 9
7695                          ;                               ;R2 = DIGIT
7696
7697  033740  121127  000060  CK DEC  CMPB    (R1),#'0        ;LESS THAN ZERO?
7698  033744  103407                  BLO     1$              ;YES -- BRANCH
7699  033746  121127  000071          CMPB    (R1),#'9                ;GREATER THAN NINE?
7700  033752  101004                  BHI     1$              ;YES -- BRANCH
7701  033754  111102                  MOVB    (R1),R2         ;GET THE CHARACTER
7702  033756  042702  000060          BIC     #'0,R2          ;STRIP AWAY THE ASCII
7703  033762  005720                  TST     (R0)+           ;ADJUST FOR RETURN
7704  033764  000200          1$      RTS     R0              ;RETURN
7705
7706                          ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
7707                          ;DETERMINE WHAT IT IS
7708                          ;CALL
7709                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
7710                          ;       JSR     R0,@#CK CHR     ;CHECK CHARACTER
7711                          ;       RETURN  ADR1            ;UNKNOWN CHARACTER
7712                          ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
7713                          ;       RETURN  ADR3            ;SLASH * (R1)=ADR+1
7714                          ;       RETURN  ADR4            ;COMMA * (R1)=ADR+1
7715                          ;       RETURN  ADR5            ;PERIOD * (R1)=ADR+1
7716                          ;       RETURN  ADR6            ;DIGIT BETWEEN 0 AND 9
7717                          ;                               ;R2 = DIGIT * (R1)=ADR+1
7718
7719  033766  105711          CK CHR  TSTB    (R1)            ;"CARRIAGE RETURN"?
7720  033770  001420                  BEQ     4$              ;YES -- BRANCH
7721  033772  121127  000057          CMPB    (R1),#'/        ;"SLASH"?
7722  033776  001414                  BEQ     3$              ;YES -- BRANCH
7723  034000  121127  000054          CMPB    (R1),#',        ;"COMMA"?
7724  034004  001410                  BEQ     2$              ;YES -- BRANCH
7725  034006  121127  000056          CMPB    (R1),#'         ;"PERIOD"?
7726  034012  001404                  BEQ     1$              ;YES -- BRANCH
7727  034014  004037  033740          JSR     R0,@#CK DEC     ;"DIGIT"?
7728  034020  000406                  BR      5$              ;NO -- BRANCH
7729  034022  005720                  TST     (R0)+           ;DIGIT BETWEEN 0-9
7730  034024  005720          1$      TST     (R0)+           ;PERIOD
7731  034026  005720          2$      TST     (R0)+           ;COMMA
7732  034030  005720          3$      TST     (R0)+           ;SLASH
7733  034032  005720          4$      TST     (R0)+           ;CARRIAGE RETURN
7734  034034  005201                  INC     R1              ;MOVE POINTER TO NEXT CHARACTER
7735  034036  011000          5$      MOV     (R0),R0         ;UNKNOWN CHARACTER
7736  034040  000200                  RTS     R0              ;RETURN
7737
7738                          ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
7739                           CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2
```

```
7740                                    ,CALL
7741                                    ,       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
7742                                    ,       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
7743                                    ,       JSR     R0,@#CK.DIG     ;CHECK DIGITS
7744                                    ,       RETURN  ADR1            ;ILLEGAL CHARACTER -- R2=?
7745                                    ,       RETURN  ADR2            ,INPUT NUMBER TO LARGE -- R2=?
7746                                    ;       RETURN  ADR3            ;"COMMA" -- R2 = NUMBER
7747                                    ,       RETURN  ADR4            ,"PERIOD" -- R2 = NUMBER
7748                                    ;       RETURN  ADR5            ;"PERIOD-PERIOD" -- R2 = NUMBER
7749
7750  034042  010446                    CK DIG  MOV     R4,-(SP)        ,SAVE R4
7751  034044  010346                            MOV     R3,-(SP)        ,SAVE R3
7752  034046  010246                            MOV     R2,-(SP)        ,SAVE THE MAX. SIZE ON THE STACK
7753  034050  005002                            CLR     R2              ,START WITH 0
7754  034052  005003                            CLR     R3
7755  034054  005004                            CLR     R4
7756  034056  004037  033766                    JSR     R0,@#CK CHR     ,CHECK ONE CHARACTER
7757  034062  034212                            8$                      ,ILLEGAL CHARACTER
7758  034064  034212                            8$                      ,CARRIAGE RETURN
7759  034066  034212                            8$                      ,"/"
7760  034070  034212                            8$                      ,","
7761  034072  034212                            8$                      ," "
7762  034074  034076                            1$                      ,DIGIT 0-9
7763  034076  006303                    1$      ASL     R3              ,2
7764  034100  010346                            MOV     R3,-(SP)        ,SAVE *2
7765  034102  006303                            ASL     R3              ,4
7766  034104  006303                            ASL     R3              ,8
7767  034106  062603                            ADD     (SP)+,R3        ,(*8)+(*2)=*10
7768  034110  060203                            ADD     R2,R3           ,UPDATE THE INPUT NUMBER
7769  034112  004037  033766                    JSR     R0,@#CK CHR     ,CHECK ONE CHARACTER
7770  034116  034212                            8$                      ,ILLEGAL CHARACTER
7771  034120  034132                            9$                      ,CARRIAGE RETURN
7772  034122  034212                            8$                      ,"/"
7773  034124  034140                            3$                      ,","
7774  034126  034136                            2$                      ;" "
7775  034130  034076                            1$                      ,DIGIT 0-9
7776  034132  005301                    9$      DEC     R1              ;BACKUP THE CHARACTER POINTER
7777  034134  000401                            BR      3$              ,CONTINE
7778  034136  005724                    2$      TST     (R4)+           ,"PERIOD"
7779  034140  005724                    3$      TST     (R4)+           ;"COMMA" OR "CR"
7780  034142  004037  033766                    JSR     R0,@#CK CHR     ,CHECK ONE CHARACTER
7781  034146  034212                            8$                      ,ILLEGAL CHARACTER
7782  034150  034202                            6$                      ,CARRIAGE RETURN
7783  034152  034212                            8$                      ,"/"
7784  034154  034212                            8$                      ,","
7785  034156  034162                            4$                      ;" "
7786  034160  034172                            5$                      ,DIGIT 0-9
7787  034162  005724                    4$      TST     (R4)+           ,"PERIOD-PERIOD"
7788  034164  105711                            TSTB    (R1)            ,"CR"?
7789  034166  001405                            BEQ     6$              ,YES--BRANCH
7790  034170  000410                            BR      8$
7791  034172  126127  177776  000054    5$      CMPB    -2(R1),#',      ,WAS CHARACTER BEFORE THE DIGIT A COMMA?
7792  034200  001004                            BNE     8$              ,NO--EXIT
7793  034202  020316                    6$      CMP     R3,(SP)         ,INPUT TO LARGE?
7794  034204  101001                            BHI     7$              ,YES -- BRANCH
7795  034206  060400                            ADD     R4,R0           ,ADJUST RETURN ADDRESS
```

```
7796  034210  005720                  7$      TST     (R0)+
7797  034212  010302                  8$      MOV     R3,R2           ;NUMBER TO R2
7798  034214  005726                          TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
7799  034216  012603                          MOV     (SP)+,R3        ;RESTORE R3
7800  034220  012604                          MOV     (SP)+,R4        ;RESTORE R4
7801  034222  011000                          MOV     (R0),R0         ;GET RETURN ADDRESS
7802  034224  000200                          RTS     R0              ;RETURN
7803
7804                                  ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
7805                                  ;AND FORMS AN OCTAL NUMBER IN R2
7806                                  ;CALL
7807                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
7808                                  ;       JSR     R0,@#CK.NUM     ;GO FORM THE NUMBER
7809                                  ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
7810                                  ;       RETURN  ADR2            ;"COMMA" OR "CR"--R2=NUMBER
7811                                  ;       RETURN  ADR3            ;"PERIOD"--R2=NUMBER
7812                                  ;       RETURN  ADR4            ;"PERIOD-PERIOD"--R2=NUMBER
7813
7814  034226  010346                  CK NUM  MOV     R3,-(SP)        ;SAVE R3
7815  034230  005003                          CLR     R3              ;START NUMBER AT ZERO
7816  034232  004037  033712                  JSR     R0,@#CK OCT     ;OCTAL DIGIT?
7817  034236  000440                          BR      6$              ;NO--BRANCH
7818  034240  005201                  1$      INC     R1              ;MOVE TO NEXT CHARACTER
7819  034242  006303                          ASL     R3              ;FOR THE OCTAL NUMBER IN R3
7820  034244  103435                          BCS     6$              ;DON'T LET IT GET TO BIG
7821  034246  006303                          ASL     R3
7822  034250  103433                          BCS     6$
7823  034252  006303                          ASL     R3
7824  034254  103431                          BCS     6$
7825  034256  060203                          ADD     R2,R3
7826  034260  004037  033712                  JSR     R0,@#CK OCT     ;IS THIS AN OCTAL DIGIT?
7827  034264  000401                          BR      2$              ;NO--FIND OUT WHAT IT IS
7828  034266  000764                          BR      1$              ;YES--MAKE IT PART OF THE NUMBER
7829  034270  010302                  2$      MOV     R3,R2           ;SAVE THE OCTAL NUMBER
7830  034272  005003                          CLR     R3              ;START WITH ZERO INDEX
7831  034274  004037  033766                  JSR     R0,@#CK CHR     ;CHECK ONE CHARACTER
7832  034300  034340                          6$                      ;ILLEGAL CHARACTER
7833  034302  034330                          5$                      ;CARRIAGE RETURN
7834  034304  034340                          6$                      ;"/"
7835  034306  034330                          5$                      ;","
7836  034310  034314                          3$                      ;"."
7837  034312  034340                          6$                      ;DIGIT 0-9
7838  034314  005723                  3$      TST     (R3)+           ;"PERIOD"
7839  034316  121127  000056                  CMPB    (R1),#'         ;"PERIOD-PERIOD"?
7840  034322  001002                          BNE     5$              ;NO--BRANCH
7841  034324  005201                          INC     R1              ;YES--ADVANCE THE POINTER
7842  034326  005723                  4$      TST     (R3)+           ;"PERIOD-PERIOD"
7843  034330  005723                  5$      TST     (R3)+           ;"COMMA"
7844  034332  105711                          TSTB    (R1)            ;"CR"?
7845  034334  001001                          BNE     6$              ;NO--BRANCH
7846  034336  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
7847  034340  012603                  6$      MOV     (SP)+,R3        ;RESTORE R3
7848  034342  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
7849  034344  000200                          RTS     R0              ;RETURN
```

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

                                 SBTTL  SINGLE/DUAL PORT RH11/RP04/5/6 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 "2"
                                        ;RPERRS   = RPDS1
                                        ;RPERRS+2 = RPER1
                                        ;RPERRS+4 = RPER2
                                        ;RPERRS+6 = RPER3

034346  000000  000000  000000  RPERRS   WORD   0,0,0,0
034354  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

034356      000                 DRVACT   BYTE   0               ;DRIVE 0
034357      000                          BYTE   0               ;DRIVE 1
034360      000                          BYTE   0               ;DRIVE 2
034361      000                          BYTE   0               ;DRIVE 3
034362      000                          BYTE   0               ;DRIVE 4
034363      000                          BYTE   0               ;DRIVE 5
034364      000                          BYTE   0               ;DRIVE 6
034365      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

034366      000                 DRVSTA:  BYTE   0               ;DRIVE 0
034367      000                          BYTE   0               ;DRIVE 1
034370      000                          BYTE   0               ;DRIVE 2
034371      000                          BYTE   0               ;DRIVE 3
034372      000                          BYTE   0               ;DRIVE 4
034373      000                          BYTE   0               ;DRIVE 5
034374      000                          BYTE   0               ;DRIVE 6
034375      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 RP04
                                        ;DRVTYP=2 IF DRIVE IS RP05
                                        ;DRVTYP=4 IF DRIVE IS RP06
                                        ;DRVTYP=-1 IF NOT RP04/5/6
```

```
7906  034376  000                DRVTYP  .BYTE   0               ;DRIVE 0
7907  034377  000                        .BYTE   0               ;DRIVE 1
7908  034400  000                        BYTE    0               ;DRIVE 2
7909  034401  000                        BYTE    0               ;DRIVE 3
7910  034402  000                        BYTE    0               ,DRIVE 4
7911  034403  000                        BYTE    0               ;DRIVE 5
7912  034404  000                        BYTE    0               ;DRIVE 6
7913  034405  000                        BYTE    0               ;DRIVE 7
7914
7915                             ,TABLE OF DUAL PORT INITIALIZATION INDICATORS
7916                                     ,DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
7917                                     ,DPINT<0 IF INITIALIZATION IS IN PROGRESS
7918
7919  034406  000                DPINT   BYTE    0               ,DRIVE 0
7920  034407  000                        BYTE    0               ,DRIVE 1
7921  034410  000                        BYTE    0               ,DRIVE 2
7922  034411  000                        BYTE    0               ,DRIVE 3
7923  034412  000                        BYTE    0               ,DRIVE 4
7924  034413  000                        .BYTE   0               ,DRIVE 5
7925  034414  000                        BYTE    0               ,DRIVE 6
7926  034415  000                        BYTE    0               ,DRIVE 7
7927
7928                             ,TABLE OF PENDING DUAL PORT REQUESTS
7929                                     ,DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
7930                                     ,DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
7931
7932  034416  000                DPRQS   BYTE    0               ;DRIVE 0
7933  034417  000                        BYTE    0               ,DRIVE 1
7934  034420  000                        BYTE    0               ,DRIVE 2
7935  034421  000                        BYTE    0               ,DRIVE 3
7936  034422  000                        BYTE    0               ,DRIVE 4
7937  034423  000                        BYTE    0               ,DRIVE 5
7938  034424  000                        BYTE    0               ,DRIVE 6
7939  034425  000                        BYTE    0               ,DRIVE 7
7940
7941                            ,TRANSFER WAIT FLAG (TRNSWT=1 WORD)
7942                                    ,THIS IS A ONE WORD QUEUE  IT WILL CONTAIN THE ADDRESS OF
7943                                    ;"DPB" OF THE I/O OPERATION
7944
7945  034426  000000             TRNSWT  .WORD   0
7946
7947                            ,SEARCH WAIT KEYS (SRCHWT=1 WORD)
7948                                    ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
7949                                    ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
7950                                    .REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE
7951                                    ,EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0
7952
7953  034430  000000             SRCHWT  .WORD   0
7954
7955                            ,RP04/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
7956                                    ,ACTDRV=0 IF DRIVER IS INACTIVE
7957                                    ,ACTDRV>0 IF DRIVER IS ACTIVE
7958
7959  034432  000                ACTDRV  BYTE    0
7960
7961                            ,SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
```

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

034433  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

034434  000                     ULDFLG   BYTE   0               ;DRIVE 0
034435  000                              BYTE   0               ;DRIVE 1
034436  000                              BYTE   0               ;DRIVE 2
034437  000                              BYTE   0               ;DRIVE 3
034440  000                              BYTE   0               ;DRIVE 4
034441  000                              BYTE   0               ;DRIVE 5
034442  000                              BYTE   0               ;DRIVE 6
034443  000                              BYTE   0               ;DRIVE 7

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

034444  000                     LACNT    BYTE   0               ;DRIVE 0
034445  000                              BYTE   0               ;DRIVE 1
034446  000                              BYTE   0               ;DRIVE 2
034447  000                              BYTE   0               ;DRIVE 3
034450  000                              BYTE   0               ;DRIVE 4
034451  000                              BYTE   0               ;DRIVE 5
034452  000                              BYTE   0               ;DRIVE 6
034453  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

034454  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

034456  000000                  SEEKFG   WORD   0

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

034460  177777                  TIMER    .WORD  -1       ;DRIVE 0
034462  177777                           WORD   -1       ;DRIVE 1
034464  177777                           WORD   -1       ;DRIVE 2
034466  177777                           WORD   -1       ;DRIVE 3
034470  177777                           WORD   -1       ;DRIVE 4
034472  177777                           WORD   -1       ;DRIVE 5
```

```
 8018  034474  177777                  .WORD   -1      ,DRIVE 6
 8019  034476  177777                  .WORD   -1      ;DRIVE 7
 8020
 8021                                ,DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
 8022                                        ,DTUW<0 IF NO DATA TRANSFER UNDERWAY
 8023                                        ,DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
 8024
 8025  034500  177777                DTUW    WORD    -1
 8026
 8027                                ,ATTENTION BITS TABLE (ATABIT=8 BYTES)
 8028                                        ,THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
 8029                                        ,ATTENTION BIT
 8030
 8031  034502     001                ATABIT  BYTE    1       ;DRIVE 0
 8032  034503     002                        BYTE    2       ,DRIVE 1
 8033  034504     004                        BYTE    4       ,DRIVE 2
 8034  034505     010                        BYTE    10      ,DRIVE 3
 8035  034506     020                        BYTE    20      ,DRIVE 4
 8036  034507     040                        BYTE    40      ,DRIVE 5
 8037  034510     100                        BYTE    100     ,DRIVE 6
 8038  034511     200                        BYTE    200     ,DRIVE 7
 8039
 8040                                ,RP04/5/6 TO RH11 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
 8041                                ,CALLING IT FATAL (MCPEMX=1 WORD)
 8042
 8043  034512  000003                MCPEMX  WORD    3
 8044
 8045                                ,STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RP04/5/6),
 8046                                ,RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5))
 8047
 8048  034514  176700                RPADR   WORD    176700
 8049  034516  000254  000240        RPVEC   WORD    254,5*32
 8050
 8051                                ,MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
 8052
 8053  034522  000004                MXLACT  WORD    4
 8054                                ,MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
 8055
 8056  034524  001000                MXDLTA  WORD    8 *64
 8057                                ,MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
 8058
 8059  034526  000200                MNDLTA  WORD    2*64
 8060                                ,MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
 8061
 8062  034530  000005                MXWNDW  WORD    5
 8063
 8064                                ,DEFINITIONS OF THE RH11/RP04/5/6 ADDRESS INDEXES
 8065
 8066          000000                RPCS1=0                 ,CONTROL AND STATUS REGISTER #1 (DRIVE REG  00)
 8067          000002                RPWC=2                  ,WORD COUNT REGISTER (NOT A DRIVE REG)
 8068          000004                RPBA=4                  ,UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
 8069          000006                RPDA=6                  ,DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG  05)
 8070          000010                RPCS2=10                ,CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
 8071          000012                RPDS1=12                ,DRIVE STATUS REGISTER (DRIVE REG 01)
 8072          000014                RPER1=14                ,ERROR REGISTER #1 (DRIVE REG  02)
 8073          000016                RPAS=16                 ,ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG  04)
```

```
 8074            000020                  RPLA=20                         ;LOOK AHEAD REGISTER (DRIVE REG. 07)
 8075            000022                  RPDB=22                         ;DATA BUFFER REGISTER (NOT A DRIVE REG )
 8076            000024                  RPMR=24                         ;MAINTAINABILITY REGISTER (DRIVE REG  03)
 8077            000026                  RPDT=26                         ;DRIVE TYPE REGISTER (DRIVE REG. 06)
 8078            000030                  RPSN=30                         ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
 8079            000032                  RPOF=32                         ;OFFSET REGISTER (DRIVE REG. 11)
 8080            000034                  RPCA=34                         ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG  12)
 8081            000036                  RPCC=36                         ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG  13)
 8082            000040                  RPER2=40                        ;ERROR REGISTER #2 (DRIVE REG. 14)
 8083            000042                  RPER3=42                        ;ERROR REGISTER #3 (DRIVE REG. 15)
 8084            000044                  RPEC1=44                        ;ECC POSITION REGISTER (DRIVE REG. 16)
 8085            000046                  RPEC2=46                        ;ECC PATTERN REGISTER (DRIVE REG.  17)
 8086
 8087                                    ;RH11/RP04/5/6 DRIVER INITIALIZATION CODE
 8088                                            ;THIS ROUTINE WILL DETERMINE WHICH RP04/5/6 DRIVES ARE
 8089                                            ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
 8090                                            ;TO THE PROPER STATE FOR EACH DRIVE.
 8091                                            ;NOTE  THIS ROUTINE CALLS DRVINT
 8092                                    ;
 8093                                    ;CALL
 8094                                    ;
 8095                                    ;       JSR     PC,RPINIT
 8096                                    ;       RETURN
 8097                                    ;
 8098                                    ;NOTE  THE 'P' OR 'L' CLOCK MUST BE STARTED
 8099                                    ;
 8100  034532  104412                    RPINIT  SAVREG                  ;SAVE R0 - R5
 8101  034534  013746  177776                    MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
 8102  034540  012737  000240  177776            MOV     #(5*32 ),@#PS   ;CHANGE THE PRIORITY TO 5
 8103  034546  004737  042516                    JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
 8104  034552  012701  034346                    MOV     #RPERRS,R1      ;FIRST ADDRESS TO BE CLEARED
 8105  034556  012702  034456                    MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
 8106  034562  005021                  1$        CLR     (R1)+           ;CLEAR
 8107  034564  020102                            CMP     R1,R2           ;ARE WE DONE?
 8108  034566  101775                            BLOS    1$              ;BRANCH IF NO
 8109  034570  012702  034500                    MOV     #DTUW,R2        ;LAST ADDRESS
 8110  034574  012721  177777            2$      MOV     #-1,(R1)+       ;INITIALIZE
 8111  034600  020102                            CMP     R1,R2           ;DONE?
 8112  034602  101774                            BLOS    2$              ;LOOP IF NO
 8113  034604  005037  034366                    CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
 8114  034610  005037  034370                    CLR     DRVSTA+2
 8115  034614  005037  034372                    CLR     DRVSTA+4
 8116  034620  005037  034374                    CLR     DRVSTA+6
 8117  034624  013703  034516                    MOV     RPVEC,R3        ;SETUP THE RH11/RP04/5/6 VECTOR
 8118  034630  012723  037376                    MOV     #ISR,(R3)+
 8119  034634  013713  034520                    MOV     RPVEC+2,(R3)
 8120  034640  013704  034514                    MOV     RPADR,R4        ;FIRST ADDRESS OF RH11/RP04
 8121  034644  012764  000040  000010            MOV     #BIT05,RPCS2(R4)  ;MASSBUS INIT
 8122  034652  005001                            CLR     R1              ;START WITH DRIVE 0
 8123  034654  004037  034744            3$      JSR     R0,DRVINT       ;INIT THE DRIVE
 8124  034660  000401                            BR      4$              ;'DVA' NOT SET OR PARITY ERROR
 8125  034662  000402                            BR      5$              ;NORMAL RETURN
 8126  034664  105061  034366            4$      CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
 8127  034670  005201                    5$      INC     R1              ;GO TO NEXT DRIVE
 8128  034672  042701  177770                    BIC     #^C7,R1         ;MASK OUT UNUSED BITS
 8129  034676  001366                            BNE     3$              ;BR IF MORE DRIVES TO GO
```

```
8130  034700  012701  000007              MOV     #7,R1           ;START WITH DRIVE 7
8131  034704  005037  177776              CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
8132  034710  105761  034406      6$      TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
8133  034714  001405                      BEQ     8$              ;BR NOT WAITING
8134  034716  004737  042152              JSR     PC,SET IE       ;SET INTERRUPT
8135  034722  105761  034406      7$      TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
8136  034726  001375                      BNE     7$              ;BR IF NOT
8137  034730  005301                      DEC     R1              ;GO TO THE NEXT DRIVE
8138  034732  100366                      BPL     6$              ;CHECK NEXT DRIVE
8139  034734  012637  177776              MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
8140  034740  104413                      RESREG                  ;RESTORE R0 - R5
8141  034742  000207                      RTS     PC              ;BYE-BYE
8142
8143                                  ;DRIVE INITILIZATION ROUTINE
8144                                  ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
8145                                  ;AN RP04/5/6  IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
8146                                  ;IS SET TO A "1"  THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
8147                                  ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
8148                                  ;DRVSTA IS SET TO THE PROPER CONDITION
8149                                  ;CALL
8150                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
8151                                  ;       MOV     RPADR,R4        ;UNIBUS ADDRESS OF RH11/RP04/5/6 (RPCS1)
8152                                  ;       JSR     R0,DRVINT       ;CALLED BY A JSR
8153                                  ;       RETURN1                 ;ERROR OCCURRED (PARITY)
8154                                  ;       RETURN2                 ;NORMAL RETURN
8155                                  ;
8156
8157  034744  010546              DRVINT  MOV     R5,-(SP)        ;SAVE R5
8158  034746  105061  034366              CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
8159  034752  105061  034376              CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
8160  034756  105061  034434              CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
8161  034762  010164  000010              MOV     R1,RPCS2(R4)    ;SELECT A DRIVE
8162  034766  112764  000111  000000      MOVB    #111,RPCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
8163  034774  032764  010000  000010      BIT     #BIT12,RPCS2(R4) ;NONEXISTENT DRIVE?
8164  035002  001403                      BEQ     1$              ;NO---BRANCH
8165  035004  004737  042152              JSR     PC,SET IE       ;GO SET "IE" WITHOUT A "TRE"
8166  035010  000520                      BR      6$              ;LEAVE THIS ROUTINE
8167  035012  105061  034366      1$      CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
8168  035016  032764  004000  000000      BIT     #BIT11,RPCS1(R4)  ;SEE IF DRIVE AVAILABLE
8169  035024  001514                      BEQ     7$              ;BR IF DRIVE NOT AVAILABLE
8170  035026  004037  041462              JSR     R0,RD RP        ;READ THE DRIVE TYPE REG
8171  035032  000026                      RPDT
8172  035034  035276                      8$                      ;ERROR RETURN ADDRESS
8173  035036  012605                      MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
8174  035040  112761  000001  034376      MOVB    #1,DRVTYP(R1)   ;SET RP04 INDICATOR
8175  035046  022705  020020              CMP     #20020,R5       ;IS IT A SINGLE PORT RP04?
8176  035052  001431                      BEQ     2$              ;BRANCH IF YES
8177  035054  022705  024020              CMP     #24020,R5       ;IS IT A DUAL PORT RP04?
8178  035060  001426                      BEQ     2$              ;BR IF YES
8179  035062  112761  000002  034376      MOVB    #2,DRVTYP(R1)   ;SET RP05 INDICATOR
8180  035070  022705  020021              CMP     #20021,R5       ;SINGLE PORT RP05 ?
8181  035074  001420                      BEQ     2$              ;BR IF YES
8182  035076  022705  024021              CMP     #24021,R5       ;DUAL PORT RP05 ?
8183  035102  001415                      BEQ     2$              ;BR IF YES
8184  035104  112761  000004  034376      MOVB    #4,DRVTYP(R1)   ;SET RP06 INDICATOR
8185  035112  022705  020022              CMP     #20022,R5       ;SINGLE PORT RP06 ?
```

```
 8186  035116  001407                          BEQ     2$                ;BR IF YES
 8187  035120  022705  024022                  CMP     #24022,R5         ;DUAL PORT RP06 ?
 8188  035124  001404                          BEQ     2$                ;BR IF YES
 8189  035126  112761  177777  034376          MOVB    #-1,DRVTYP(R1)    ,SET INDICATOR TO 'OTHER'
 8190  035134  000446                          BR      6$                ;EXIT
 8191  035136  012746  000121          2$      MOV     #121,-(SP)        ,DO A "READ-IN PRESET"
 8192  035142  004037  041642                  JSR     R0,WRT RP
 8193  035146  000000                          RPCS1
 8194  035150  035276                          8$
 8195  035152  012746  010000                  MOV     #BIT12,-(SP)      ,SET FMT22=1
 8196  035156  004037  041642                  JSR     R0,WRT RP
 8197  035162  000032                          RPOF
 8198  035164  035276                          8$
 8199  035166  004037  041462                  JSR     R0,RD RP          ,READ RPDS1
 8200  035172  000012                          RPDS1
 8201  035174  035276                          8$
 8202  035176  012605                          MOV     (SP)+,R5          ,AND SAVE IT IN R5
 8203  035200  100015                          BPL     4$                ,BRANCH IF ATA=0
 8204  035202  116164  034502  000016          MOVB    ATABIT(R1),RPAS(R4) ,CLEAR ATTENTION BIT
 8205  035210  004037  041462                  JSR     R0,RD RP          ,FIND OUT WHY ATA=1
 8206  035214  000014                          RPER1
 8207  035216  035276                          8$
 8208  035220  006126                          ROL     (SP)+             ,IS IT UNSAFE?
 8209  035222  100004                          BPL     4$                ,BR IF NOT
 8210  035224  112761  177777  034366          MOVB    #-1,DRVSTA(R1)    ,SET UNSAFE INDICATOR
 8211  035232  000407                          BR      6$                ,EXIT
 8212  035234  005105                  4$      COM     R5                ,CHECK MOL, DPR, DRY, AND VV
 8213  035236  042705  167077                  BIC     # C<BIT12!BIT08!BIT07!BIT06>,R5
 8214  035242  001003                          BNE     6$                ,BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
 8215  035244  112761  000001  034366          MOVB    #1,DRVSTA(R1)     ,SET DRIVE STATUS TO ONLINE
 8216  035252  005720                  6$      TST     (R0)+             ,STEP OVER THE ERROR RETURN
 8217  035254  000410                          BR      8$                ,EXIT
 8218  035256  006301                  7$      ASL     R1                ,CHANGE INDEX TO ADDRESS WORDS
 8219  035260  012761  003720  034460          MOV     #2000,TIMER(R1)   ,START 2 SEC TIMER
 8220  035266  006201                          ASR     R1                ,RESTORE R1
 8221  035270  105161  034406                  COMB    DPINT(R1)
 8222  035274  005720                          TST     (R0)+
 8223  035276  012605                  8$      MOV     (SP)+,R5          ,RESTORE R5
 8224  035300  000200                          RTS     R0                ,EXIT
 8225
 8226                                  ,REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
 8227                                  ,
 8228                                  ,CALL
 8229                                  ,
 8230                                  ,       JSR     R0,@#RP04         ,CALL THE RP04/5/6 DRIVER
 8231                                  ,       PNTADR                    ,ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
 8232                                  ,       RETURN1                   ,RETURN HERE IF QUEUE IS FULL
 8233                                  ,       RETURN2                   ,RETURN HERE IF REQUEST IS IN QUEUE OR THERE
 8234                                                                    ,IS AN ERROR CONDITION
 8235
 8236  035302  013746  177776          RP04    MOV     @#PS,-(SP)        ,SAVE THE CALLING STATUS
 8237  035306  013737  034520  177776          MOV     RPVEC+2,@#PS      ,DON'T ALLOW ANY RP04/5/6 INTERRUPTS
 8238  035314  112737  000001  034432          MOVB    #1,ACTDRV         ,SET "ACTIVE DRIVER" FLAG
 8239  035322  104412                          SAVREG                    ,SAVE R0 - R5
 8240  035324  011002                          MOV     (R0),R2           ,PICKUP THE DRIVE PARAMETER BLOCK POINTER
 8241  035326  005062  000016                  CLR     16(R2)            ,CLEAR THE STATUS/ERROR INDICATOR
```

```
8242  035332  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
8243  035334  013704  034514                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF RPCS1
8244  035340  105761  034366                  TSTB    DRVSTA(R1)      ,CHECK DRIVES STATUS
8245  035344  003014                          BGT     1$              ;BRANCH IF ONLINE
8246  035346  105761  034434                  TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
8247  035352  001036                          BNE     3$              ;BRANCH IF YES
8248  035354  105761  034406                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
8249  035360  001042                          BNE     5$              ;BR IF YES
8250  035362  004037  034744                  JSR     R0,DRVINT       ,GO INIT. THE DRIVE
8251  035366  000434                          BR      4$              ;ERROR RETURN
8252  035370  105761  034366                  TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
8253  035374  003445                          BLE     6$              ;BR IF NOT
8254  035376  105761  034416          1$      TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
8255  035402  001031                          BNE     5$              ,BR IF YES
8256  035404  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
8257  035410  004037  042614                  JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
8258  035414  000460                          BR      9$              ,QUEUE IS FULL
8259  035416  122762  000103  000002          CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
8260  035424  001003                          BNE     2$              ,BR IF NO
8261  035426  112761  177777  034434          MOVB    #-1,ULDFLG(R1)  ,SET THE "UNLOAD IN QUEUE" FLAG
8262  035434  105761  034356          2$      TSTB    DRVACT(R1)      ,IS THIS DRIVE ACTIVE?
8263  035440  001043                          BNE     8$              ,BR IF YES
8264  035442  004737  035574                  JSR     PC,OPT          ,CALL THE OPTIMIZER
8265  035446  000440                          BR      8$
8266  035450  012762  120000  000016  3$      MOV     #BIT15!BIT13,16(R2) ,SET THE "UNLOAD IN QUEUE" ERROR FLAG
8267  035456  000434                          BR      8$              ,EXIT
8268  035460  004737  036704          4$      JSR     PC,C17          ,GO HANDLE THE PARITY ERROR
8269  035464  000431                          BR      8$
8270  035466  004037  042614          5$      JSR     R0,DRVQUE       ,PUT REQUEST IN QUEUE
8271  035472  000431                          BR      9$              ,QUEUE IS FULL
8272  035474  032714  000100                  BIT     #BIT06,(R4)     ,IS 'IE' SET ALREADY ?
8273  035500  001023                          BNE     8$              ,BR IF IT IS
8274  035502  004737  042152                  JSR     PC,SET IE       ,SET INTERRUPT
8275  035506  000420                          BR      8$              ,RETURN, REQUEST IN QUEUE
8276  035510  105761  034366          6$      TSTB    DRVSTA(R1)      ,SEE IF DRIVE OFFLINE OR UNSAFE
8277  035514  002412                          BLT     7$              ,BR IF UNSAFE
8278  035516  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE ERROR INDICATOR
8279  035524  105761  034376                  TSTB    DRVTYP(R1)      ,SEE IF OFFLINE OR NONEXISTENT
8280  035530  001007                          BNE     8$              ,BR IF OFFLINE
8281  035532  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;REPORT DRIVE NONEXISTENT
8282  035540  000403                          BR      8$              ,GO TO EXIT
8283  035542  012762  110000  000016  7$      MOV     #BIT15!BIT12,16(R2)  ;DRIVE IS UNSAFE
8284  035550  104413                  8$      RESREG                  ,RESTORE R0 - R5
8285  035552  005720                          TST     (R0)+           ,SETUP FOR NORMAL RETURN
8286  035554  000401                          BR      10$             ,FINISH UP, THEN EXIT
8287  035556  104413                  9$      RESREG                  ;RESTORE R0 - R5
8288  035560  005720                  10$     TST     (R0)+           ,CORRECT THE RETURN ADDRESS
8289  035562  105037  034432                  CLRB    ACTDRV          ,CLEAR "ACTIVE DRIVER" FLAG
8290  035566  012637  177776                  MOV     (SP)+,@#PS      ;RETURN "PS" TO USER LEVEL
8291  035572  000200                          RTS     R0              ;RETURN TO CALLER
8292
8293                                  ,OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
8294                                  ,
8295                                  ,CALL
8296                                  ,       MOV     #DRVNUM,R1      ,DRIVE NUMBER TO R1
8297                                  ,       JSR     PC,OPT          ,SETUP A COMMAND
```

```
 8298
 8299  035574  104412                    OPT     SAVREG                  ;SAVE R0 - R5
 8300  035576  013746  177776                    MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
 8301  035602  146137  034502  034430            BICB    ATABIT(R1),SRCHWT       ;CLEAR "SEARCH WAIT" KEY
 8302  035610  004737  042670                    JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
 8303  035614  005702                            TST     R2              ;IS THERE A REQUEST IN QUEUE?
 8304  035616  001505                            BEQ     7$              ;NO--BRANCH TO EXIT
 8305  035620  032764  004000  000000            BIT     #BIT11,RPCS1(R4)  ;IS DVA STILL SET ?
 8306  035626  001407                            BEQ     10$             ;BR IF NOT
 8307  035630  032764  000100  000012            BIT     #BIT6,RPDS1(R4)  ;IS VV SET ?
 8308  035636  001003                            BNE     10$             ;BR IF IT IS
 8309  035640  004037  034744            9$      JSR     R0,DRVINT       ;SEE IF DRIVE STILL ONLINE ?
 8310  035644  000470                            BR      6$              ;PARITY OR 'DVA' NOT SET
 8311  035646  105761  034366            10$     TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
 8312  035652  003014                            BGT     1$              ;YES--BRANCH
 8313  035654  004737  042712                    JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
 8314  035660  012762  140000  000016            MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
 8315  035666  105761  034366                    TSTB    DRVSTA(R1)      ;IS DRIVE UNSAFE ?
 8316  035672  100064                            BPL     8$              ;BR TO EXIT IF NOT
 8317  035674  012762  110000  000016            MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
 8318  035702  000460                            BR      8$              ;BRANCH TO EXIT
 8319  035704  012746  000111            1$      MOV     #111,-(SP)      ;LOAD COMMAND ONTO THE STACK
 8320  035710  004037  041642                    JSR     R0,WRT RP       ,LOAD THE REGISTER
 8321  035714  000000                            RPCS1                   ,REGISTER INCREMENT
 8322  035716  036026                            6$                      ,ERROR RETURN ADDRESS
 8323  035720  032714  004000                    BIT     #BIT11,(R4)     ,DRIVE AVAILABLE ?
 8324  035724  001427                            BEQ     5$              ,BR IF NOT
 8325  035726  122762  000150  000002            CMPB    #150,2(R2)      ,IS THE REQUEST FOR I/O?
 8326  035734  002403                            BLT     2$              ,YES--BRANCH
 8327  035736  004737  036270                    JSR     PC,C14          ,CALL THE COMMAND INITIATOR
 8328  035742  000440                            BR      8$              ,BRANCH TO EXIT
 8329  035744  005737  034500            2$      TST     DTUW            ,DATA TRANSFER UNDERWAY?
 8330  035750  002012                            BGE     4$              ,YES--GO START A SEARCH
 8331  035752  005737  034456                    TST     SEEKFG          ,DO IMPLIED SEEKS?
 8332  035756  100404                            BMI     3$              ,YES---BRANCH
 8333  035760  004037  037240                    JSR     R0,LA           ,NO--DO LOOK AHEAD
 8334  035764  000427                            BR      8$              ,RETURN HERE ON A PARITY ERROR
 8335  035766  000403                            BR      4$              ,GO START A SEARCH
 8336  035770  004737  036054            3$      JSR     PC,C11          ,START A DATA TRANSFER
 8337  035774  000423                            BR      8$
 8338  035776  004737  036162            4$      JSR     PC,C13          ,START A SEARCH
 8339  036002  000420                            BR      8$              ,GO TO THE EXIT
 8340  036004  112761  177777  034416    5$      MOVB    #-1,DPRQS(R1)   ,SET PORT REQUEST INDICATOR
 8341  036012  010103                            MOV     R1,R3           ,SET UP TO ADDRESS WORDS
 8342  036014  006303                            ASL     R3              ,CONVERT TO WORD INDEX
 8343  036016  012763  023420  034460            MOV     #10000,TIMER(R3)  ,START 10 SEC TIMER
 8344  036024  000402                            BR      7$              ,EXIT
 8345  036026  004737  036704            6$      JSR     PC,C17          ,PROCESS THE PARITY ERROR
 8346  036032  032714  000100            7$      BIT     #BIT06,(R4)     ,SEE IF 'IE' ALREADY SET
 8347  036036  001002                            BNE     8$              ,BR IF SET
 8348  036040  004737  042152                    JSR     PC,SET IE       ,SET "IE" WITHOUT A "TRE"
 8349  036044  012637  177776            8$      MOV     (SP)+,@#PS      ;RESTORE PROC  STATUS
 8350  036050  104413                            RESREG                  ,RESTORE R0 - R5
 8351  036052  000207                            RTS     PC
 8352
 8353                                    ,COMMAND INITIATOR
```

```
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                        ;       MOV     #DPB,R2         ;ADDRESS OF DPB
                                        ;       JSR     PC,C1?          ;C1?= C11,C13, OR C14
                                        ;                               ;WHERE:
                                        ;                               ;C11=DATA TRANSFER
                                        ;                               ;C12=SEARCH REQUESTED BY DATA XFER
                                        ;                               ;C14=NOT DATA TRANSFER

036054  004737  042712        C11     JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
036060  010237  034426                MOV     R2,TRNSWT       ;PUT REQ.  IN TRANSFER WAIT QUEUE
036064  010203                        MOV     R2,R3           ;DPB ADDRESS TO R3
036066  013704  034514                MOV     RPADR,R4        ,RPCS1 ADDRESS
036072  010164  000010                MOV     R1,RPCS2(R4)    ,SELECT DRIVE
036076  062703  000004                ADD     #4,R3           ,DESIRED WORD COUNT
036102  062704  000002                ADD     #2,R4           ,RPWC ADDRESS
036106  012324                        MOV     (R3)+,(R4)+     ,LOAD WORD COUNT
036110  012324                        MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
036112  012346                        MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
036114  004037  041642                JSR     R0,WRT RP       ,CALL THE LOAD(WRITE) ROUTINE
036120  000006                        RPDA                    ,INDEX OF REGISTER TO LOAD
036122  036704                        C17                     ,ERROR RETURN ADDRESS
036124  012346                        MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
036126  004037  041642                JSR     R0,WRT RP
036132  000034                        RPCA
036134  036704                        C17
036136  016246  000002                MOV     2(R2),-(SP)     ,LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
036142  004037  041642                JSR     R0,WRT RP
036146  000000                        RPCS1
036150  036704                        C17
036152  010137  034500                MOV     R1,DTUW         ,SET "DATA TRANSFER UNDERWAY"
036156  000137  036646                JMP     C15
036162  013704  034514        C13.    MOV     RPADR,R4        ,RPCS1 ADDRESS
036166  010164  000010                MOV     R1,RPCS2(R4)    ,SELECT DRIVE
036172  016246  000012                MOV     12(R2),-(SP)    ,DESIRED CYLINDER ADDRESS
036176  004037  041642                JSR     R0,WRT RP
036202  000034                        RPCA
036204  036704                        C17
036206  116203  000010                MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
036212  163703  034530                SUB     MXWNDW,R3       ,BACKUP BY MAX  SEARCH FOR I/O WINDOW
036216  002002                        BGE     1$
036220  062703  000026                ADD     #22 ,R3
036224  010346                1$.     MOV     R3,-(SP)        ,COMBINE THE ADJUSTED SECTOR WITH
036226  116266  000011  000001        MOVB    11(R2),1(SP)    ,THE DESIRED TRACK
036234  004037  041642                JSR     R0,WRT RP       ,LOAD DESIRED TRACK & SECTOR
036240  000006                        RPDA
036242  036704                        C17
036244  012746  000131                MOV     #131,-(SP)      ,START A SEARCH
036250  004037  041642                JSR     R0,WRT RP
036254  000000                        RPCS1
036256  036704                        C17
036260  156137  034502  034430        BISB    ATABIT(R1),SRCHWT       ,SET "SEARCH WAIT" KEY
036266  000567                        BR      C15
036270  013704  034514        C14     MOV     RPADR,R4        ,RPCS1 ADDRESS
036274  010164  000010                MOV     R1,RPCS2(R4)    ,SELECT DRIVE
```

```
8410  036300  116203  000002              MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
8411  036304  122703  000131              CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
8412  036310  001007                      BNE     1$              ;BRANCH IF NO
8413  036312  016246  000010              MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
8414  036316  004037  041642              JSR     R0,WRT RP
8415  036322  000006                      RPDA
8416  036324  036704                      C17
8417  036326  000403                      BR      2$              ;GO LOAD CYLINDER
8418  036330  122703  000105      1$      CMPB    #105,R3         ;IS IT A SEEK COMMAND
8419  036334  001007                      BNE     3$              ;BRANCH IF NO
8420  036336  016246  000012      2$      MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
8421  036342  004037  041642              JSR     R0,WRT RP
8422  036346  000034                      RPCA
8423  036350  036704                      C17
8424  036352  000546                      BR      C16
8425  036354  122703  000115      3$      CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
8426  036360  001013                      BNE     4$              ;BR IF NO
8427  036362  004037  041462              JSR     R0,RD RP        ;MERGE THE OFFSET VALUE INTO RPOF
8428  036366  000032                      RPOF                    ;BUT DON'T CHANGE THE UPPER
8429  036370  036704                      C17
8430  036372  116216  000001              MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
8431  036376  004037  041642              JSR     R0,WRT RP       ;REGISTER (RPOF)
8432  036402  000032                      RPOF
8433  036404  036704                      C17
8434  036406  000530                      BR      C16             ;GO START THE COMMAND
8435  036410  122703  000107      4$      CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
8436  036414  001525                      BEQ     C16             ;BRANCH IF YES
8437  036416  122703  000117              CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
8438  036422  001522                      BEQ     C16             ;BRANCH IF YES
8439  036424  122703  000103              CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
8440  036430  001016                      BNE     5$              ;BRANCH IF NO
8441  036432  112761  000001  034356      MOVB    #1,DRVACT(P1)   ;SET THE DRIVE ACTIVE INDICATOR
8442  036440  105061  034366              CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
8443  036444  112761  000001  034434      MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
8444  036452  010346                      MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
8445  036454  004037  041642              JSR     R0,WRT RP
8446  036460  000000                      RPCS1
8447  036462  036704                      C17
8448  036464  000207                      RTS     PC              ;RETURN TO USER
8449  036466  122703  000143      5$      CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
8450  036472  001014                      BNE     6$              ;BRANCH IF NO
8451  036474  004037  041462              JSR     R0,RD RP        ;READ THE OFFSET REGISTER
8452  036500  000032                      RPOF
8453  036502  036704                      C17
8454  036504  116266  000001  000001      MOVB    1(R2),1(SP)     ;COMBINE "FMT22","EC ", AND "HCI"
8455  036512  004037  041642              JSR     R0,WRT RP       ;LOAD "FMT22", "ECI", AND/OR "HCI"
8456  036516  000032                      RPOF
8457  036520  036704                      C17
8458  036522  000436                      BR      12$
8459  036524  122703  000141      6$      CMPB    #141,R3         ;IS IT A "GET REGISTER" COMMAND?
8460  036530  001023                      BNE     10$             ;BRANCH IF NO
8461  036532  016203  000006      7$      MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
8462                                                              ;TO PUT THE REGISTER(S)
8463  036536  116237  000010  036554      MOVB    10(R2),9$       ;INIT  THE INDEX FOR THE FIRST REG
8464  036544  116205  000011              MOVB    11(R2),R5       ;INDEX OF LAST REG  TO MOVE
8465  036550  004037  041462      8$      JSR     R0,RD RP         READ RP04/5/6 REGISTER
```

```
8466  036554  000000                  9$      RPCS1                   ;INDEX OF REG.  TO READ
8467  036556  036704                          C17
8468  036560  012623                          MOV     (SP)+,(R3)+     ,GET THE CONTENTS OF RH11/RP04/5/6 REG
8469  036562  023705  036554                  CMP     9$,R5           ,LAST REG.  BEEN READ?
8470  036566  001414                          BEQ     12$             ;GET OUT IF YES
8471  036570  062737  000002  036554          ADD     #2,9$           ;INCREASE THE INDEX BY 2
8472  036576  000764                          BR      8$              ;LOOP--MORE TO READ
8473  036600  122703  000145          10$     CMPB    #145,R3         ;IS IT A "SELECT DRIVE" COMMAND?
8474  036604  001405                          BEQ     12$             ;BRANCH IF YES
8475  036606  010346                  11$     MOV     R3,-(SP)        ,LOAD THE COMMAND
8476  036610  004037  041642                  JSR     R0,WRT RP
8477  036614  000000                          RPCS1
8478  036616  036704                          C17
8479  036620  004737  042712          12$     JSR     PC,POPQUE       ,REMOVE REQ  FROM QUEUE
8480  036624  052762  000200  000016          BIS     #BIT07,16(R2)   ,SET THE "DONE" BIT
8481  036632  005737  034454                  TST     SAVEFG          ,SAVE THE RH11/RP04/5/6 REGISTERS?
8482  036636  100002                          BPL     13$             ,BRANCH IF NO
8483  036640  004737  042034                  JSR     PC,SVRH11       ,YES--GO SAVE THE REGISTERS
8484  036644  000207                  13$     RTS     PC              ,RETURN TO USER
8485  036646  006301                  C15     ASL     R1
8486  036650  012761  001750  034460          MOV     #1000 ,TIMER(R1)        ,SET A ONE SECOND TIMER
8487  036656  006201                          ASR     R1
8488  036660  112761  000001  034356          MOVB    #1,DRVACT(R1)   ,SET THE DRIVE ACTIVE
8489  036666  000207                          RTS     PC              ,RETURN TO THE USER
8490  036670  010346                  C16     MOV     R3,-(SP)        ,LOAD THE COMMAND
8491  036672  004037  041642                  JSR     R0,WRT RP
8492  036676  000000                          RPCS1
8493  036700  036704                          C17
8494  036702  000761                          BR      C15
8495  036704  032764  010000  000010  C17     BIT     #BIT12,RPCS2(R4)  ,DRIVE NON-EXISTENT ?
8496  036712  001034                          BNE     C18             ,BR IF YES
8497  036714  005702                  1$      TST     R2              ,ANYTHING IN QUEUE ?
8498  036716  001405                          BEQ     C17B            ,BR IF NOT
8499  036720  012762  104000  000016          MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
8500  036726  004737  042034                  JSR     PC,SVRH11       ,GO SAVE THE RH11/RP04/5/6 REGISTERS
8501  036732  012746  000111          C17B    MOV     #111,-(SP)       DO A "DRIVE CLEAR"
8502  036736  004037  041642                  JSR     R0,WRT.RP
8503  036742  000000                          RPCS1
8504  036744  037004                          C18
8505  036746  004737  042574                  JSR     PC,EMPTYQ       ,EMPTY THE QUEUE
8506  036752  105061  034434                  CLRB    ULDFLG(R1)      ,CLEAR THE UNLOAD IN QUEUE FLAG
8507  036756  105061  034356                  CLRB    DRVACT(R1)      ,DRIVE IS IDLE
8508  036762  020137  034500                  CMP     R1,DTUW         ,IF THIS DRIVE HAD AN I/O REQUEST
8509  036766  001005                          BNE     1$              ,IN PROGRESS CLEAR ALL OF THE FLAGS
8510  036770  005037  034426                  CLR     TRNSWT
8511  036774  012737  177777  034500          MOV     #-1,DTUW
8512  037002  000207                  1$      RTS     PC
8513  037004  104412                  C18.    SAVREG                  ;SAVE R0 - R5
8514  037006  032764  010000  000010          BIT     #BIT12,RPCS2(R4)  ,IS 'NED' SET ?
8515  037014  001002                          BNE     1$              ,BR IF YES
8516  037016  005001                          CLR     R1
8517  037020  005003                          CLR     R3
8518  037022  105761  034356          1$      TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
8519  037026  001443                          BEQ     5$              ,BRANCH IF NO
8520  037030  013702  034426                  MOV     TRNSWT,R2       ,GET THE "TRANSFER WAIT" QUEUE
8521  037034  020137  034500                  CMP     R1,DTUW         ,DID THIS DRIVE HAVE AN I/O IN PROGRESS?
```

```
8522  037040  001402                        BEQ     2$                ;BRANCH IF YES
8523  037042  004737  042670                JSR     PC,GETREQ         ;GET THE DPB POINTER
8524  037046  005702              2$        TST     R2                ;QUEUE ENTRY FOR DRIVE ?
8525  037050  001415                        BEQ     4$                ;BR IF NOT
8526  037052  032764  010000  000010        BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
8527  037060  001404                        BEQ     3$                ;BR IF NOT
8528  037062  012762  100002  000016        MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
8529  037070  000405                        BR      4$                ,CONTINUE
8530  037072  012762  102000  000016  3$    MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
8531  037100  004737  042034                JSR     PC,SVRH11         ;SAVE RH11/RP04/5/6 REGISTERS
8532  037104  012763  177777  034460  4$    MOV     #-1,TIMER(R3)     ;STOP THE TIMER
8533  037112  105061  034356                CLRB    DRVACT(R1)        ;SET "DRIVE ACTIVE" TO IDLE
8534  037116  020137  034500                CMP     R1,DTUW           ;IS THIS DRIVE SETUP FOR A TRANSFER
8535  037122  001005                        BNE     5$                ,BR IF NOT
8536  037124  012737  177777  034500        MOV     #-1,DTUW          ;RESET THE INDICATOR
8537  037132  005037  034426                CLR     TRNSWT            ;CLEAR THE TRANSFER QUEUE
8538  037136  105061  034434          5$    CLRB    ULDFLG(R1)        ;CLEAR UNLOAD FLAG
8539  037142  032764  010000  000010        BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
8540  037150  001021                        BNE     6$                ,BR IF YES
8541  037152  005201                        INC     R1                ;MOVE TO THE NEXT DRIVE
8542  037154  062703  000002                ADD     #2,R3
8543  037160  042701  177770                BIC     # C7,R1
8544  037164  001316                        BNE     1$                ,BRANCH IF MORE DRIVES
8545  037166  012737  177777  034500        MOV     #-1,DTUW          ,NO DATA TRANSFERS UNDERWAY
8546  037174  005037  034426                CLR     TRNSWT            ;CLEAR THE 'TRANSFER WAIT' QUEUE
8547  037200  004737  042516                JSR     PC,CLRQUE         ,CLEAR ALL OF THE REQUEST QUEUES
8548  037204  012764  000040  000010        MOV     #BIT05,RPCS2(R4)  ,DO A MASSBUS INIT
8549  037212  000406                        BR      7$                ,CONTINUE
8550  037214  004737  042574          6$    JSR     PC,EMPTYQ         ,CLEAR THE DRIVE'S QUEUE
8551  037220  105061  034366                CLRB    DRVSTA(R1)        ,SET DRIVE TO OFFLINE
8552  037224  105061  034376                CLRB    DRVTYP(R1)        ,CLEAR THE DRIVE TYPE INDICATOR
8553  037230  004737  042152          7$    JSR     PC,SET IE         ,SET "IE" WITHOUT "TRE"
8554  037234  104413                        RESREG                    ,RESTORE R0 - R5
8555  037236  000207                        RTS     PC                ,RETURN
8556
8557                                  ,LOOK AHEAD ROUTINE
8558                                  ,
8559                                  ,CALL
8560                                  ,       MOV     #DRVNUM,R1        ,DRIVE NUMBER
8561                                  ,       MOV     #DPB,R2           ,POINT TO DPB
8562                                  ,       JSR     R0,LA             ,GO CHECK THE WINDOW
8563                                  ,       RETURN1                   ,ERROR RETURN
8564                                  ,       RETURN2                   ,START A SEARCH
8565                                  ;       RETURN3                   ,START A DATA TRANSFER
8566
8567  037240  013704  034514          LA      MOV     RPADR,R4          ,GET RPCS1'S ADDRESS
8568  037244  010164  000010                  MOV     R1,RPCS2(R4)      ,SELECT DRIVE
8569  037250  004037  041462                  JSR     R0,RD RP          ,READ CURRENT CYLINDER
8570  037254  000036                        RPCC
8571  037256  037370                        4$                        ;ERROR RETURN ADDRESS
8572  037260  022662  000012                CMP     (SP)+,12(R2)      ,IS CURRENT CYLINDER=DESIRED
8573                                                                    ,CYLINDER?
8574  037264  001037                        BNE     3$                ,EXIT IF NO
8575  037266  105261  034444                INCB    LACNT(R1)         ,INCREMENT THE LOOK AHEAD COUNT
8576  037272  126137  034444  034522        CMPB    LACNT(R1),MXLACT  ,EXCEED MAX?
8577  037300  003026                        BGT     2$                ,BRANCH IF YES
```

```
 8578  037302  116203  000010                  MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
 8579  037306  000303                          SWAB    R3              ;MULT. BY 64--ALIGN WITH
 8580  037310  006203                          ASR     R3              ;LOOK AHEAD REGISTER
 8581  037312  006203                          ASR     R3
 8582  037314  012737  000340  177776          MOV     #340,@#PS       ;PRIORITY LEVEL "7"
 8583  037322  004037  041462                  JSR     R0,RD RP        ,READ LOOK AHEAD REGISTER
 8584  037326  000020                          RPLA
 8585  037330  037370                          4$
 8586  037332  162603                          SUB     (SP)+,R3        ,CALCULATE THE DELTA
 8587  037334  002002                          BGE     1$
 8588  037336  062703  002600                  ADD     #(22 *64 ),R3   ;MAKE THE DELTA POSITIVE
 8589  037342  023703  034524          1$      CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
 8590  037346  002406                          BLT     3$              ;IF IT IS WITHIN THE
 8591  037350  023703  034526                  CMP     MNDLTA,R3       ,WINDOW---IF YES, ZERO
 8592  037354  002003                          BGE     3$              ,THE LOOK AHEAD COUNT
 8593  037356  105061  034444          2$      CLRB    LACNT(R1)       ,AND TAKE THE I/O EXIT
 8594  037362  005720                          TST     (R0)+
 8595  037364  005720                  3$      TST     (R0)+           ,ADJUST THE RETURN ADDRESS
 8596  037366  000402                          BR      5$              ,EXIT
 8597  037370  004737  036704          4$      JSR     PC,C17          ,PROCESS THE ERROR
 8598  037374  000200                  5$      RTS     R0              ,RETURN
 8599
 8600                                  ,INTERRUPT SERVICE ROUTINE
 8601
 8602  037376  112737  000001  034432  ISR     MOVB    #1,ACTDRV       ,SET "ACTIVE DRIVER" FLAG
 8603  037404  104412                          SAVREG                  ,SAVE R0 - R5
 8604  037406  013704  034514                  MOV     RPADR,R4        ,ADDRESS OF RHSCS1
 8605  037412  013701  034500                  MOV     DTUW,R1         ,GET "DATA TRANSFER UNDERWAY" INDICATOR
 8606  037416  002403                          BLT     1$              ,BRANCH IF NO DATA TRANSFER UNDERWAY
 8607  037420  004737  037442                  JSR     PC,TD           ,CALL TRANSFER DONE
 8608  037424  000402                          BR      2$              ,EXIT
 8609  037426  004737  037602          1$      JSR     PC,SC           ,CALL SPECIAL CONDITIONS
 8610  037432  104413                  2$      RESREG                  ,RESTORE R0 - R5
 8611  037434  105037  034432                  CLRB    ACTDRV          ,CLEAR "ACTIVE DRIVER" FLAG
 8612  037440  000002                          RTI                     ,RETURN
 8613
 8614                                  ,TRANSFER DONE ROUTINE
 8615
 8616  037442  105061  034356          TD      CLRB    DRVACT(R1)      ,SET DRIVE ACTIVE INDICATOR TO IDLE
 8617  037446  012737  177777  034500          MOV     #-1,DTUW        ,NO DATA TRANSFERS UNDERWAY
 8618  037454  006301                          ASL     R1
 8619  037456  012761  177777  034460          MOV     #-1,TIMER(R1)   ,CANCEL TIMEOUT
 8620  037464  006201                          ASR     R1
 8621  037466  013702  034426                  MOV     TRNSWT,R2       ,GET "DPB" ADDRESS FROM THE
 8622  037472  005037  034426                  CLR     TRNSWT          ,TRANSFER WAIT QUEUE--CLEAR QUEUE
 8623  037476  052762  000200  000016          BIS     #BIT07,16(R2)   ,SET DONE
 8624  037504  010164  000010                  MOV     R1,RPCS2(R4)    ,SELECT THE DRIVE
 8625  037510  004037  041462                  JSR     R0,RD RP        ,TRANSFER ERROR(TRE=1)?
 8626  037514  000000                          RPCS1
 8627  037516  036704                          C17
 8628  037520  006126                          ROL     (SP)+
 8629  037522  100413                          BMI     3$              ,BR IF YES
 8630  037524  005737  034454                  TST     SAVEFG          ,SAVE THE RH11/RP04/5/6 REGISTERS?
 8631  037530  100002                          BPL     1$              ,BRANCH IF NO
 8632  037532  004737  042034                  JSR     PC,SVRH11       ,YES--SAVE THE REGISTERS
 8633  037536  004737  035574          1$      JSR     PC,OPT          ,CALL OPTIMIZER
```

```
8634  037542  000417                          BR      SC              ,CHECK OTHER DRIVES
8635  037544  012714  000113          2$      MOV     #113,(R4)       ;RELEASE THE DRIVE
8636  037550  000414                          BR      SC              ,CHECK FOR OTHER DRIVES
8637  037552  052762  100100  000016  3$      BIS     #BIT15!BIT06,16(R2) ,SET DATA ERROR FLAG
8638  037560  004737  042574                  JSR     PC,EMPTYQ       ,EMPTY THE "DRIVE'S WAIT" QUEUE
8639  037564  004737  042034                  JSR     PC,SVRH11       ,SAVE THE RH11/RP04/5/6 REGISTERS
8640  037570  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
8641  037574  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
8642  037600  000400                          BR      SC              ,CHECK FOR OTHER DRIVES
8643
8644                                  ,SPECIAL CONDITION ROUTINE
8645
8646  037602  116403  000016          SC      MOVB    RPAS(R4),R3     ,READ "RPAS"
8647  037606  001014                          BNE     2$              ,BRANCH IF ANY 'ATA' BITS SET
8648  037610  004037  041462                  JSR     R0,RD RP        ,READ CONTROL AND STATUS REGISTER
8649  037614  000000                          RPCS1
8650  037616  037004                          C18
8651  037620  106126                          ROLB    (SP)+           ,IS "IE"=1?
8652  037622  100405                          BMI     1$              ,YES, NO DRIVES TO CHECK
8653  037624  004037  042754                  JSR     R0,ES SAV       ,SAVE THE ADDRESS IN '$ESCAPE'
8654  037630  104001                          ERROR   1               ,REPORT AN ILLEGAL INTERRUPT
8655  037632  004737  042152                  JSR     PC,SET  I       ,SET INTERRUPT ENABLE
8656  037636  000207                  1$      RTS     PC              ,RETURN
8657  037640  005046                  2$      CLR     -(SP)           ,PROCESS ALL DRIVES THAT HAVE
8658  037642  110316                          MOVB    R3,(SP)         ,AN "ATA"=1
8659  037644  012703  000001                  MOV     #1,R3
8660  037650  005001                          CLR     R1
8661  037652  030316                  SC3     BIT     R3,(SP)         ,ATA=1?
8662  037654  001005                          BNE     SC5             ,YES--BRANCH
8663  037656  005201                  SC4     INC     R1              ,MOVE TO THE NEXT DRIVE
8664  037660  106303                          ASLB    R3
8665  037662  001373                          BNE     SC3             ,BRANCH IF MORE TO CHECK?
8666  037664  005726                          TST     (SP)+           ,CLEAN OFF THE STACK
8667  037666  000207                          RTS     PC              ,RETURN TO USER
8668  037670  105761  034406          SC5     TSTB    DPINT(R1)       ,INITIALIZING THE DRIVE ?
8669  037674  001402                          BEQ     1$              ,BR IF NOT
8670  037676  000137  040574                  JMP     SC13            ,PROCESS THE DRIVE
8671  037702  105761  034416          1$      TSTB    DPRQS(R1)       ,PORT REQUEST OUTSTANDING ?
8672  037706  001402                          BEQ     2$              ,BR IF NOT
8673  037710  000137  040574                  JMP     SC13            ,START THE OUTSTANDING COMMAND
8674  037714  105761  034366          2$      TSTB    DRVSTA(R1)      ,CHECK THE DRIVE STATUS
8675  037720  003025                          BGT     5$              ,BRANCH IF ONLINE
8676  037722  105761  034434                  TSTB    ULDFLG(R1)      ,UNLOAD IN PROGRESS?
8677  037726  003422                          BLE     5$              ,BRANCH IF NOT
8678  037730  004737  042670                  JSR     PC,GETREQ       ,GET DPB POINTER
8679  037734  004737  042034                  JSR     PC,SVRH11       ,SAVE THE RH11/RP04/5/6 REGISTERS
8680  037740  004737  040524                  JSR     PC,SC12         ,SAVE RPDS1, RPER1, RPER2, AND RPER3
8681                                                                  ,ALSO DO A DRIVE INIT (DRVINT)
8682  037744  105761  034366                  TSTB    DRVSTA(R1)      ,DID DRIVE COME ONLINE?
8683  037750  003416                          BLE     6$              ,NO---BRANCH
8684  037752  032737  040000  034346          BIT     #BIT14,RPERRS   ,WAS THERE AN ERROR?
8685  037760  001002                          BNE     3$              ,BR IF ERROR
8686  037762  000137  040434                  JMP     SC11            ,NO ERROR
8687  037766  013705  034350          3$      MOV     RPERRS+2,R5     ,YES -- PICKUP RPER1 AND
8688  037772  000502                          BR      SC6A            ,GO PROCESS THE ERROR
8689  037774  105761  034356          5$      TSTB    DRVACT(R1)      ,DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
```

```
 8690  040000  001033                        BNE     SC6             ,BR IF EITHER
 8691  040002  004737  040524                JSR     PC,SC12         ;SAVE RPDS1, RPER1, RPER2, AND RPER3
 8692                                                                  ;ALSO DO A DRVINT
 8693  040006  105761  034406          6$    TSTB    DPINT(R1)       ,TRYING TO INIT THE DPIVE ?
 8694  040012  001321                        BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
 8695  040014  105761  034366                TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
 8696  040020  100412                        BMI     7$              ;BR IF UNSAFE
 8697  040022  032737  020000  034354        BIT     #BIT13,RPERRS+6 ;ADDRESS PLUG CHANGED ?
 8698  040030  001013                        BNE     8$              ;BR IF YES
 8699  040032  012746  000113                MOV     #113,-(SP)      ;RELEASE COMMAND
 8700  040036  004037  041642                JSR     R0,WRT RP       ,WRITE THE COMMAND INTO RPCS1
 8701  040042  000000                        RPCS1                   ,REGISTER INDEX
 8702  040044  040404                        SC8                     ;PARITY EXIT ADDRESS
 8703  040046  011605                7$      MOV     (SP),R5         ,PICKUP (RPAS) BEFORE THE ERROR CALL
 8704  040050  004037  042754                JSR     R0,ES SAV       ,SAVE THE ADDRESS IN '$ESCAPE'
 8705  040054  104002                        ERROR   2               ,REPORT THE UNEXPECTED ATTENTION
 8706  040056  000677                        BR      SC4             ,GO CHECK FOR MORE ATA'S
 8707  040060                        8$
 8708  040060  004037  042754                JSR     R0,ES SAV       ,SAVE THE ADDRESS IN '$ESCAPE'
 8709  040064  104005                        ERROR   5               ,REPORT THE ADDRESS PLUG CHANGE
 8710  040066  000673                        BR      SC4             ,CHECK FOR MORE DRIVES
 8711  040070  006301                SC6     ASL     R1              ,SETUP TO ADDRESS WORDS
 8712  040072  012761  177777  034460        MOV     #-1,TIMER(R1)   ,STOP THE TIMER
 8713  040100  006201                        ASR     R1              ,RESTORE THE DRIVE ADDRESS
 8714  040102  004737  042670                JSR     PC,GETREQ       ,GET THE DPB POINTER FROM THE QUEUE
 8715  040106  010164  000010                MOV     R1,RPCS2(R4)    ,SELECT DRIVE
 8716  040112  004037  041462                JSR     R0,RD RP        ,READ THE RP04'S STATUS REG
 8717  040116  000012                        RPDS1
 8718  040120  040404                        SC8
 8719  040122  011605                        MOV     (SP),R5         ,AND PUT IT IN R5
 8720  040124  006126                        ROL     (SP)+           ,WAS THERE AN ERROR?
 8721  040126  100407                        BMI     1$              ,BR IF ERROR
 8722  040130  105761  034356                TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
 8723  040134  003137                        BGT     SC11            ,BR IF DRIVE ACTIVE WITH ORDER
 8724  040136  052762  100210  000016        BIS     #BIT15!BIT07!BIT03,16(R2)  ,INFORM USER OF ERROR RECOVER COMPLETION
 8725  040144  000470                        BR      SC7
 8726  040146  004037  041462          1$    JSR     R0,RD RP        ,READ ERROR REGISTER #1
 8727  040152  000014                        RPER1
 8728  040154  040404                        SC8
 8729  040156  012605                        MOV     (SP)+,R5        ,AND SAVE IT IN R5
 8730  040160  004737  042034                JSR     PC,SVRH11       ,SAVE RH11/RP04/5/6 REGISTERS
 8731  040164  012746  000111                MOV     #111,-(SP)      ,ISSUE A DRIVE CLEAR
 8732  040170  004037  041642                JSR     R0,WRT RP
 8733  040174  000000                        RPCS1
 8734  040176  040404                        SC8
 8735  040200  006105                SC6A    ROL     R5              ,WAS "UNSAFE' CONDITION =1?
 8736  040202  100406                        BMI     1$              ,BRANCH IF YES
 8737  040204  005702                        TST     R2              ,ANYTHING IN QUEUE ?
 8738  040206  001447                        BEQ     SC7             ,BR IF NOT
 8739  040210  052762  100240  000016        BIS     #BIT15!BIT07!BIT05,16(R2)       ,INFORM USER OF ERROR
 8740  040216  000443                        BR      SC7
 8741  040220  004037  041462          1$    JSR     R0,RD RP        ,READ DRIVE STATUS REG  #1
 8742  040224  000012                        RPDS1
 8743  040226  040404                        SC8
 8744  040230  011605                        MOV     (SP),R5         ,SAVE RPDS1 IN R5
 8745  040232  006126                        ROL     (SP)+           ,"ERR"=1?
```

```
8746  040234  100011                          BPL     2$              ,BR IF NO--UNSAFE CLEARED
8747  040236  112761  177777  034366          MOVB    #-1,DRVSTA(R1)  ,DRIVE IS UNSAFE
8748  040244  004737  042034                  JSR     PC,SVRH11       ,SAVE RH11/RP04/5/6 REGISTERS
8749  040250  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ,INFORM USER OF UNSAFE ERROR
8750  040256  000423                          BR      SC7
8751  040260  032705  010000          2$      BIT     #BIT12,R5       ;"MOL" = 1 ?
8752  040264  001015                          BNE     3$              ,BR IF YES
8753  040266  112761  177777  034356          MOVB    #-1,DRVACT(R1)  ,ACTIVE ERROR RECOVER
8754  040274  112761  000001  034366          MOVB    #1,DRVSTA(R1)   ,ONLINE
8755  040302  006301                          ASL     R1
8756  040304  012761  072460  034460          MOV     #30000.,TIMER(R1) ,START 30 SECOND TIMER
8757  040312  006201                          ASR     R1
8758  040314  000137  037656                  JMP     SC4
8759  040320  052762  100220  000016  3$      BIS     #BIT15!BIT07!BIT04,16(R2)       ,INFORM USER OF ERROR
8760  040326  105061  034356          SC7     CLRB    DRVACT(R1)      ,DRIVE IS IDLE
8761  040332  004737  042574                  JSR     PC,EMPTYQ       ,DUMP THE QUEUE
8762  040336  105761  034434                  TSTB    ULDFLG(R1)      ,UNLOAD IN PROGRESS OR QUEUE?
8763  040342  003002                          BGT     1$              ,BR IF NOT
8764  040344  105061  034434                  CLRB    ULDFLG(R1)      ,CLEAR UNLOAD FLAG
8765  040350  116164  034502  000016  1$      MOVB    ATABIT(R1),RPAS(R4)  ,CLEAR ATTENTION BIT
8766  040356  105761  034366                  TSTB    DRVSTA(R1)      ,IS THE DRIVE UNSAFE ?
8767  040362  100406                          BMI     2$              ,BR IF IT IS
8768  040364  012746  000113                  MOV     #113,-(SP)      ,RELEASE COMMAND
8769  040370  004037  041642                  JSR     R0,WRT RP       ,WRITE THE COMMAND INTO RPCS1
8770  040374  000000                          RPCS1                   ,REGISTER INDEX
8771  040376  040404                          SC8                     ,PARITY EXIT ADDRESS
8772  040400  000137  037656          2$      JMP     SC4             ,CHECK FOR MORE DRIVES
8773  040404  105761  034356          SC8     TSTB    DRVACT(R1)      ,IS DRIVE IDLE?
8774  040410  001405                          BEQ     1$              ,YES--BRANCH
8775  040412  004737  042670                  JSR     PC,GETREQ       ,GET DPB POINTER
8776  040416  004737  036704                  JSR     PC,C17          ,PROCESS THE PARITY ERROR
8777  040422  000402                          BR      2$              ,CONTINUE
8778  040424  004737  036732          1$      JSR     PC,C17B         ,PROCESS THE UNCORRECTABLE PARITY ERROR
8779  040430  000137  037656          2$      JMP     SC4             ,CHECK MORE DRIVES
8780  040434  105761  034434          SC11    TSTB    ULDFLG(R1)      ,"UNLOAD IN PROGRESS"?
8781  040440  003402                          BLE     1$              ,BRANCH IF NO
8782  040442  105061  034434                  CLRB    ULDFLG(R1)      ,CLEAR UNLOAD FLAG
8783  040446  105061  034356          1$      CLRB    DRVACT(R1)      ,SET DRIVE IDLE
8784  040452  136137  034502  034430          BITB    ATABIT(R1),SRCHWT  ,DOING A SEARCH OPERATION FOR
8785                                                                  ;AN I/O COMMAND?
8786  040460  001012                          BNE     2$              ,BRANCH IF YES
8787  040462  004737  042712                  JSR     PC,POPQUE       ,REMOVE REQUEST FROM QUEUE
8788  040466  052762  000200  000016          BIS     #BIT07,16(R2)   ,SET "DONE" BIT
8789  040474  005737  034454                  TST     SAVEFG          ,SAVE THE REGISTERS?
8790  040500  100002                          BPL     2$              ,BRANCH IF NO
8791  040502  004737  042034                  JSR     PC,SVRH11       ,YES--SAVE ALL OF THE RH11/RP04/5/6 REG'S
8792  040506  116164  034502  000016  2$      MOVB    ATABIT(R1),RPAS(R4)  ,CLEAR ATTENTION BIT
8793  040514  004737  035574                  JSR     PC,OPT          ,START A REQUEST
8794  040520  000137  037656                  JMP     SC4             ,CHECK FOR MORE DRIVES
8795  040524  010164  000010          SC12    MOV     R1,RPCS2(R4)    ,SELECT DRIVE
8796  040530  016437  000012  034346          MOV     RPDS1(R4),RPERRS        ,SAVE THE FOUR REGISTERS THAT
8797  040536  016437  000014  034350          MOV     RPER1(R4),RPERRS+2      ,WILL TELL US SOMETHING
8798  040544  016437  000040  034352          MOV     RPER2(R4),RPERRS+4
8799  040552  016437  000042  034354          MOV     RPER3(R4),RPERRS+6
8800  040560  004037  034744                  JSR     R0,DRVINT       ,INIT  THE STATE OF THE DRIVE
8801  040564  000401                          BR      1$              ,TAKE ERROR EXIT
```

```
                                        RTS     PC                  ;RETURN
                                1$      TST     (SP)+               ;POP PC OFF OF THE STACK
                                        BR      SC8                 ;PROCESS THE PARITY ERROR
                                SC13    ASL     R1                  ;SETUP TO ADDRESS WORDS
177777  034460                          MOV     #-1,TIMER(R1)       ;STOP THE TIMER
                                        ASR     R1                  ;
000010                                  MOV     R1,RPCS2(R4)        ;SELECT THE DRIVE
034502  000016                          MOVB    ATABIT(R1),RPAS(R4) ;CLEAR THE ATTENTION BIT
004000                                  BIT     #BIT11,(R4)         ;DRIVE AVAILABLE ?
                                        BNE     1$                  ;BR IF AVAILABLE
                                        ASL     R1
023420  034460                          MOV     #10000,TIMER(R1)    ;START 10 SEC TIMER AGAIN
                                        ASR     R1
                                        BR      3$                  ;EXIT
034406                          1$      TSTB    DPINT(R1)           ;INITIALIZING THE DRIVE ?
                                        BEQ     2$                  ;BR IF NOT
034406                                  CLRB    DPINT(R1)           ;CLEAR THE INIT INDICATOR
034744                                  JSR     R0,DRVINT           ;GO INIT THE DRIVE
                                        NOP                         ;DUMMY PARITY ERROR RETURN
034366                                  TSTB    DRVSTA(R1)          ;DRIVE ONLINE ?
                                        BGT     2$                  ;BR IF YES -- START ORDER
                                        TST     R2                  ;QUEUE ENTRY FOR THE DRIVE
                                        BEQ     3$                  ;BR IF NOT
042670                                  JSR     PC,GETREQ           ;GET DPB ADDRESS
140000  000016                          BIS     #BIT15!BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
042034                                  JSR     PC,SVRH11           ;SAVE THE REGISTERS
042574                                  JSR     PC,EMPTYQ           ;EMPTY THE REQUEST QUEUE
                                        BR      3$
034416                          2$      CLRB    DPRQS(R1)           ;CLEAR THE PORT REQUEST INDICATOR
035574                                  JSR     PC,OPT              ;START THE PENDING REQUEST
037656                          3$      JMP     SC4                 ;PROCESS OTHER DRIVES

                                ;RP04/5/6 TIMER ROUTINE
                                ;CALL
                                ;       MOV     #TIME,-(SP)         ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                ;       JSR     PC,RPTMR            ;CALL RP04,5/6 TIME ROUTINE

034432                          RPTMR   TST     ACTDRV              ;CHECK "ACTDRV & ACTSTR"
                                        BNE     4$                  ;IF NON ZERO EXIT
000001  034433                          MOVB    #1,ACTSTR           ;SET "ACTSTR"
                                        SAVREG                      ;SAVE R0 - R5
                                        CLR     R1                  ;START WITH DRIVE 0
                                        CLR     R3
034460                          1$      TST     TIMER(R3)           ;IS THE TIMER RUNNING?
                                        BLT     2$                  ;BRANCH IF NO
000002  034460                          SUB     2(SP),TIMER(R3)     ;COUNT THE INTERVAL
                                        BGT     2$                  ;BR IF NO SOFTWARE TIMEOUT
041026                                  JSR     PC,STO              ;CALL SOFTWARE TIMEOUT ROUTINE
                                        BR      3$                  ;GO TO THE EXIT
                                2$      INC     R1                  ;MOVE TO NEXT DRIVE
                                        TST     (R3)+
000010                                  CMP     #8,R1               ;OUT OF DRIVES?
                                        BGT     1$                  ;BRANCH IF NO
                                3$      RESREG                      ;RESTORE R0 - R5
034433                                  CLRB    ACTSTR              ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
                                4$      MOV     (SP)+,(SP)          ;ADJUST THE STACK
```

```
 8858  041024  000207                          RTS     PC              ;RETURN
 8859
 8860                                  ;SOFTWARE TIMEOUT ROUTINE
 8861                                  ;
 8862                                  ;NOTE  THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
 8863                                  ;       OR GREATER
 8864                                  ;
 8865                                  ;CALL   STO
 8866                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
 8867                                  ;       JSR     PC,STO          ;CALL
 8868                                  ;       RETURN
 8869
 8870  041026  010146                  STO     MOV     R1,-(SP)        ;SAVE R1
 8871  041030  010346                          MOV     R3,-(SP)        ;SAVE R3
 8872  041032  013704  034514                  MOV     RPADR,R4        ;GET ADDRESS OF 'RPCS1"
 8873  041036  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
 8874  041042  004037  041462                  JSR     R0,RD RP        ;READ "DRIVE STATUS REG"
 8875  041046  000012                          RPDS1
 8876  041050  041350                          ST05
 8877  041052  105726                          TSTB    (SP)+           ;IS "DRY"=1?
 8878  041054  100477                          BMI     ST02            ;BR IF YES
 8879  041056  105761  034406        ST01    TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
 8880  041062  001074                          BNE     ST02            ;BR IF YES
 8881  041064  105761  034416                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
 8882  041070  001071                          BNE     ST02            ;BR IF YES
 8883  041072  013702  034426                  MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
 8884  041076  020137  034500                  CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
 8885  041102  001402                          BEQ     1$              ;BRANCH IF YES
 8886  041104  004737  042670                  JSR     PC,GETREQ       ;GET DPB ADDRESS
 8887  041110  052762  101000  000016  1$      BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
 8888  041116  004737  042034                  JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
 8889  041122  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;"INIT" THE MASS BUS
 8890  041130  105061  034356                  CLRB    DRVACT(R1)      ;DRIVE IS IDLE
 8891  041134  105061  034434                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
 8892  041140  005001                          CLR     R1              ;START WITH DRIVE 0
 8893  041142  005003                          CLR     R3
 8894  041144  004037  034744          2$      JSR     R0,DRVINT       ;INIT  THIS DRIVE
 8895  041150  000477                          BR      ST05            ;PARITY ERROR RETURN
 8896  041152  105761  034356                  TSTB    DRVACT(R1)      ;DRIVE IDLE BEFORE THE INIT ?
 8897  041156  001414                          BEQ     4$              ;YES--BRANCH
 8898  041160  013702  034426                  MOV     TRNSWT,R2       ;GET TRANSFER WAIT QUEUE
 8899  041164  023701  034500                  CMP     DTUW,R1         ;WAS THERE I/O ON THIS DRIVE?
 8900  041170  001402                          BEQ     3$              ;YES--BRANCH
 8901  041172  004737  042670                  JSR     PC,GETREQ       ;GET THE DPB POINTER FROM QUEUE
 8902  041176  052762  100400  000016  3$      BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT
 8903  041204  105061  034356                  CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE TO IDLE
 8904  041210  105061  034434          4$      CLRB    ULDFLG(R1)      ;NO UNLOAD
 8905  041214  012763  177777  034460          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
 8906  041222  005723                          TST     (R3)+           ;UPDATE THE INDEX
 8907  041224  005201                          INC     R1              ;INCREMENT THE DRIVE NUMBER
 8908  041226  022701  000010                  CMP     #8 ,R1          ;LAST DRIVE BEEN CHECKED?
 8909  041232  003344                          BGT     2$              ;NO--LOOP
 8910  041234  012737  177777  034500          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
 8911  041242  005037  034426                  CLR     TRNSWT          ;CLEAR TRANSFER WAIT QUEUE
 8912  041246  004737  042516                  JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
 8913  041252  000500                          BR      ST09            ;EXIT
```

```
 8914  041254  116405  000016          ST02    MOVB    RPAS(R4),R5     ;READ ATTENTION REG
 8915  041260  136105  034502                  BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
 8916  041264  001017                          BNE     ST03            ;YES--BRANCH
 8917  041266  105761  034406                  TSTB    DPINT(R1)       ,TRYING TO INTIALIZE THE DRIVE ?
 8918  041272  001031                          BNE     ST06            ;BR IF YES - DRIVE NOT ONLINE
 8919  041274  105761  034416                  TSTB    DPRQS(R1)       ,OUTSTANDING PORT REQUEST FOR THE DRIVE ?
 8920  041300  001045                          BNE     ST07            ;BR IF YES - NO RESPONSE TO REQUEST
 8921  041302  020137  034500                  CMP     R1,DTUW         ,DATA TRANSFER UNDERWAY FOR THIS DRIVE
 8922  041306  001263                          BNE     ST01            ,BR IF NO
 8923  041310  004037  041462                  JSR     R0,RD RP        ,YES--CHECK "RDY"
 8924  041314  000000                          RPCS1
 8925  041316  041350                          ST05
 8926  041320  105726                          TSTB    (SP)+
 8927  041322  100255                          BPL     ST01            ,BR IF "RDY"=0
 8928  041324  105761  034406          ST03    TSTB    DPINT(R1)       ,INITIALIZING THE DRIVE ?
 8929  041330  001003                          BNE     1$               BR IF INIT PENDING
 8930  041332  105761  034416                  TSTB    DPRQS(R1)       ,PORT REQUEST PENDING ?
 8931  041336  001446                          BEQ     ST09            ,BR IF NOT
 8932  041340  012763  177777  034460  1$      MOV     #-1,TIMER(R3)   ,STOP THE TIMER
 8933  041346  000442                          BR      ST09            ,EXIT
 8934  041350  004737  037004          ST05    JSR     PC,C18          ,GO HANDLE THE PARITY ERROR
 8935  041354  000437                          BR      ST09
 8936  041356  105061  034406          ST06    CLRB    DPINT(R1)       ,CLEAR THE INITIALIZE INDICATOR
 8937  041362  105061  034366                  CLRB    DRVSTA(R1)      ,SET UNIT OFFLINE
 8938  041366  012763  177777  034460          MOV     #-1,TIMER(R3)  ,STOP THE TIMER
 8939  041374  004737  042670                  JSR     PC,GETREQ       ,GET THE DPB ADDRESS
 8940  041400  005702                          TST     R2              ,REQUEST IN QUEUE ?
 8941  041402  001424                          BEQ     ST09            ,BR IF NOT
 8942  041404  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
 8943  041412  000414                          BR      ST08            ,FINISH
 8944  041414  012763  177777  034460  ST07    MOV     #-1,TIMER(R3)   ,STOP THE TIMER
 8945  041422  105061  034416                  CLRB    DPRQS(R1)       ,CLEAR PORT REQUEST INDICATOR
 8946  041426  004737  042670                  JSR     PC,GETREQ       ,GET DPB ADDRESS
 8947  041432  005702                          TST     R2              ,QUEUE ENTRY FOR DRIVE ?
 8948  041434  001407                          BEQ     ST09            ,BR IF NONE
 8949  041436  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)  ,INFORM USER OF PORT REQUEST ERROR
 8950  041444  004737  042574          ST08    JSR     PC,EMPTYQ       ,CLEAR THE QUEUE FOR THE DRIVE
 8951  041450  004737  042034                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
 8952  041454  012603                  ST09    MOV     (SP)+,R3        ,RESTORE R3
 8953  041456  012601                          MOV     (SP)+,R1        ,RESTORE R1
 8954  041460  000207                          RTS     PC              ,RETURN
 8955
 8956                                  ,ROUTINE TO READ A RH11/RP04/5/6 REGISTER
 8957                                  ,
 8958                                  ,CALL
 8959                                  ,       JSR     R0,RD RP        ,GO READ A REGISTER
 8960                                  ,       INDEX                   ,REG  INDEX FROM BASE
 8961                                  ,       ERRADR                  ,ERROR ADDRESS--PROCESS ERROR STARTING
 8962                                  ,                               ,AT THIS ADDRESS
 8963                                  ,       RETURN                  ,CONTENTS OF REG  IS ON THE STACK
 8964                                  ,
 8965  041462  013737  034512  041630  RD RP   MOV     MCPEMX,RD RP2   ,MAX  RETRYS ALLOWED
 8966  041470  011646                          MOV     (SP),-(SP)      ,SAVE R0 FOR RETURN
 8967  041472  013737  034514  041506          MOV     RPADR,RD ADR    ,FORM THE DESIRED ADDRESS
 8968  041500  062037  041506                  ADD     (R0)+,RD ADR    ,USING THE BASE AND THE INDEX
 8969  041504  013727                  RD RP1  MOV     @(PC)+,(PC)+    ,READ THE DESIRED REGISTER OF THE RP04
```

```
8970  041506  000000                    RD ADR.  .WORD   0               ,ADDRESS IS FORMED HERE
8971  041510  000000                    RD WRD   .WORD   0               ;REG. CONTENTS PUT HERE
8972  041512  013766  041510  000002            MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
8973  041520  013746  034514                    MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
8974  041524  062716  000010                    ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
8975  041530  032736  010000                    BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
8976  041534  001037                            BNE     RD RP3          ;BR IF DRIVE NON-EXISTENT
8977  041536  017746  172752                    MOV     @RPADR,-(SP)    ;READ RPCS1
8978  041542  032716  020000                    BIT     #BIT13,(SP)     ,DID MCPE SET?
8979  041546  001002                            BNE     1$              ;BRANCH IF YES
8980  041550  022620                            CMP     (SP)+,(R0)+     ,ADJUST FOR RETURN
8981  041552  000432                            BR      RD RP4          ,EXIT
8982  041554                            1$
8983  041554  004037  042754                    JSR     R0,ES SAV       ,SAVE THE ADDRESS IN '$ESCAPE'
8984  041560  104003                            ERROR   3               ,REPORT "MCPE" ERROR
8985  041562  005737  034500                    TST     DTUW            ,DATA TRANSFER UNDERWAY?
8986  041566  100405                            BMI     2$              ;NO--BRANCH
8987  041570  032716  040000                    BIT     #BIT14,(SP)     ,NO--"TRE"=1?
8988  041574  001402                            BEQ     2$              ,NO--BRANCH
8989  041576  005726                            TST     (SP)+           ,YES--CLEAN OFF THE STACK AND
8990  041600  000415                            BR      RD RP3          ,TAKE THE FATAL ERROR EXIT
8991  041602  052716  040000            2$      BIS     #BIT14,(SP)     ,CLEAR "MCPE" BY SENDING A "1" TO "TRE"
8992  041606  000316                            SWAB    (SP)            ,POSITION BEFORE WRITING
8993  041610  013737  034514  041624            MOV     RPADR,3$        ,FORM ADDRESS OF HIGH BYTE
8994  041616  005237  041624                    INC     3$
8995  041622  112637                            MOVB    (SP)+,@(PC)+    ,WRITE THE HIGH BYTE OF RPCS1
8996  041624  000000            3$               WORD    0               ,ADDRESS STORAGE.
8997  041626  005327                            DEC     (PC)+           ,EXCEEDED MAX  RETRYS
8998  041630  000003            RD RP2           WORD    3
8999  041632  002324                            BGE     RD RP1          ,BRANCH IF NO
9000  041634  011000            RD RP3  MOV     (R0),R0                 ,FATAL ERROR EXIT
9001  041636  012616                    MOV     (SP)+,(SP)
9002  041640  000200            RD RP4  RTS     R0
9003
9004                                    ,ROUTINE TO WRITE A REGISTER
9005                                    ,
9006                                    ,CALL
9007                                    ,       MOV     DATA,-(SP)      ,DATA TO BE LOADED ON THE STACK
9008                                    ,       JSR     R0,WRT RP       ,CALL THE ROUTINE TO LOAD(WRITE) THE REG
9009                                    ,       INDEX                   ,INDEX OF THE REGISTER TO BE LOADED
9010                                    ,       ERRADR                  ,ADDRESS TO RETURN TO ON AN ERROR
9011                                    ,       RETURN                  ,ERROR FREE RETURN
9012
9013  041642  013737  034512  042016  WRT RP  MOV     MCPEMX,WRT R2   ,MAX RETRYS ALLOWED
9014  041650  016637  000002  041730          MOV     2(SP),WRT WD    ,SAVE THE WORD TO WRITE
9015  041656  012616                    MOV     (SP)+,(SP)      ;ADJUST THE STACK
9016  041660  012037  041732                    MOV     (R0)+,WRT AD    ,GET INDEX OF REGISTER TO BE WRITTEN
9017  041664  001015                            BNE     1$              ;BRANCH IF NOT RPCS1
9018  041666  122737  000150  041730          CMPB    #150,WRT WD     ,IS THE COMMAND FOR DATA TRANSFERS?
9019  041674  002411                            BLT     1$              ,YES--DON'T GET THE OLD A16 & A17, & PSEL
9020  041676  004037  041462                    JSR     R0,RD RP        ,NO---COMBINE A16&A17, & PSEL WITH
9021  041702  000000                    RPCS1                           ,THE COMMAND BEFORE SENDING IT TO
9022  041704  042024                    WRT R3                          ,THE RH11/RP04
9023  041706  000316                    SWAB    (SP)
9024  041710  042716  177770                    BIC     # C7,(SP)
9025  041714  112637  041731                    MOVB    (SP)+,WRT WD+1
```

```
 9026  041720  063737  034514  041732  1$:       ADD     RPADR,WRT AD    ;FORM THE ADDRESS OF THE DISK REG
 9027  041726  012737                   WRT.R1:   MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
 9028  041730  000000                   WRT.WD:   .WORD   0               ;WORD TO WRITE GOES HERE
 9029  041732  000000                   WRT AD.   .WORD   0               ;ADDRESS IS FORMED HERE
 9030  041734  013746  034514                     MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
 9031  041740  062716  000010                     ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
 9032  041744  032736  010000                     BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
 9033  041750  001025                             BNE     WRT R3          ;BR IF DRIVE NON-EXISTENT
 9034  041752  004037  041462                     JSR     R0,RD RP        ;CHECK FOR PARITY ERROR ON WRITE
 9035  041756  000014                             RPER1
 9036  041760  042024                             WRT R3
 9037  041762  032726  000010                     BIT     #BIT03,(SP)+
 9038  041766  001420                             BEQ     WRT R4          ,BRANCH IF "PAR=0"
 9039  041770  016037  177776  042002             MOV     -2(R0),1$       ,PICKUP THE INDEX
 9040  041776  004037  041462                     JSR     R0,RD RP        ,READ THE REG.
 9041  042002  000000                   1$         WORD    0               ,REG. INDEX
 9042  042004  042024                             WRT R3                  ,RETURN TO THIS ADDRESS ON ERROR
 9043  042006  004037  042754                     JSR     R0,ES SAV       ,SAVE THE ADDRESS IN '$ESCAPE'
 9044  042012  104004                             ERROR   4               ,REPORT THE PARITY ON WRITE ERROR
 9045  042014  005327                             DEC     (PC)+           ,DECREMENT THE ERROR COUNT
 9046  042016  000003                   WRT R2     WORD    3               ,RETRY COUNTER
 9047  042020  002342                             BGE     WRT R1          ,TRY AGAIN IF NOT FINISHED
 9048  042022  005726                             TST     (SP)+           ,CLEAN OFF THE STACK
 9049  042024  011000                   WRT R3    MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
 9050  042026  000401                             BR      WRT R5          ,EXIT
 9051  042030  005720                   WRT R4    TST     (R0)+           ,ADJUST FOR ERROR FREE EXIT
 9052  042032  000200                   WRT R5    RTS     R0
 9053
 9054                                   ,ROUTINE TO SAVE THE RH11/RP04/5/6 REGISTERS AS PER DPB+14
 9055                                   ,
 9056                                   ,CALL
 9057                                   ,         MOV     #DPBNUM,R2      ,DPB POINTER TO R2
 9058                                   ,         JSR     PC,SVRH11       ,SAVE THE DRIVES REG'S
 9059
 9060  042034  104412                   SVRH11    SAVREG                  ,SAVE R0 - R5
 9061  042036  005702                             TST     R2              ,QUEUE ENTRY FOR THE DRIVE ?
 9062  042040  001430                             BEQ     4$              ,BR IF NONE
 9063  042042  013704  034514                     MOV     RPADR,R4
 9064  042046  111264  000010                     MOVB    (R2),RPCS2(R4)  ,SELECT DRIVE
 9065  042052  016203  000014                     MOV     14(R2),R3       ,GET THE ERROR TABLE POINTER
 9066  042056  001433                             BEQ     6$              ,EXIT IF NO ADDRESS
 9067  042060  005037  042114                     CLR     3$              ,COUNTER & POINTER
 9068  042064  023727  042114  000022   1$        CMP     3$,#RPDB        ,REACHED THE BUFFER REGISTER ?
 9069  042072  001006                             BNE     2$              ,BR IF NOT
 9070  042074  032764  000200  000010             BIT     #BIT07,RPCS2(R4)  ;'OR' SET ?
 9071  042102  001002                             BNE     2$              ,BR IF SET
 9072  042104  005023                             CLR     (R3)+           ;STORE RPDB AS ZEROES
 9073  042106  000405                             BR      4$              ,CONTINUE
 9074  042110  004037  041462            2$       JSR     R0,RD RP        ;READ THE SELECTED REGISTER
 9075  042114  000000                   3$        .WORD   0               ;REGISTER INDEX
 9076  042116  042142                             5$                      ,ERROR RETURN ADDRESS
 9077  042120  012623                             MOV     (SP)+,(R3)+     ,STORE THE REGISTER CONTENTS
 9078  042122  023727  042114  000046   4$        CMP     3$,#RPEC2       ,REACHED THE END ?
 9079  042130  001406                             BEQ     6$              ,BR IF YES
 9080  042132  062737  000002  042114             ADD     #2,3$           ,INCREMENT THE REGISTER INDEX
 9081  042140  000751                             BR      1$              ,CONTINUE READING THE REGISTERS
```

```
 9082  042142  004737  036704          5$:     JSR     PC,C17          ;PROCESS THE UNCORRECTABLE PARITY ERROR
 9083  042146  104413                  6$      RESREG                  ;RESTORE R0 - R5
 9084  042150  000207                          RTS     PC              ;RETURN
 9085
 9086                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
 9087                                  ;CALL
 9088                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 9089                                  ;       JSR     PC,SET.IE       ;SET "IE"
 9090                                  ;       RETURN
 9091
 9092  042152  010446                  SET IE  MOV     R4,-(SP)        ;SAVE R4
 9093  042154  013704  034514                  MOV     RPADR,R4        ;PICKUP ADDRESS OF RPCS1
 9094  042160  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
 9095  042164  011446                          MOV     (R4),-(SP)      ;READ RPCS1
 9096  042166  052716  040000                  BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
 9097  042172  000316                          SWAB    (SP)            ;ADJUST FOR DATO
 9098  042174  112714  000100                  MOVB    #BIT06,(R4)     ;SET "IE"
 9099  042200  032764  010000  000010          BIT     #BIT12,RPCS2(R4) ;IS "NED"=1?
 9100  042206  001002                          BNE     1$              ;YES--CLEAR "TRE"
 9101  042210  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 9102  042212  000402                          BR      2$
 9103  042214  112664  000001          1$      MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
 9104  042220  012604                  2$      MOV     (SP)+,R4        ;RESTORE R4
 9105  042222  000207                          RTS     PC              ;RETURN TO CALLER
 9106
 9107                                  ;QUEUE COUNT
 9108  042224      000                 QCNT      BYTE  0               ;DRIVE 0
 9109  042225      000                         BYTE    0               ;DRIVE 1
 9110  042226      000                         BYTE    0               ;DRIVE 2
 9111  042227      000                         BYTE    0               ;DRIVE 3
 9112  042230      000                         BYTE    0               ;DRIVE 4
 9113  042231      000                         BYTE    0               ;DRIVE 5
 9114  042232      000                         BYTE    0               ;DRIVE 6
 9115  042233      000                         BYTE    0               ;DRIVE 7
 9116
 9117                                  ;QUEUE INPUT POINTERS
 9118
 9119  042234  042316                  QINPT     WORD  QDRV0           ;DRIVE 0
 9120  042236  042336                          WORD    QDRV1           ;DRIVE 1
 9121  042240  042356                          WORD    QDRV2           ;DRIVE 2
 9122  042242  042376                          WORD    QDRV3           ;DRIVE 3
 9123  042244  042416                          WORD    QDRV4           ;DRIVE 4
 9124  042246  042436                          WORD    QDRV5           ;DRIVE 5
 9125  042250  042456                          WORD    QDRV6           ;DRIVE 6
 9126  042252  042476                          WORD    QDRV7           ;DRIVE 7
 9127
 9128                                  ;QUEUE OUTPUT POINTERS
 9129
 9130  042254  042316                  QOUTPT    WORD  QDRV0           ;DRIVE 0
 9131  042256  042336                          WORD    QDRV1           ;DRIVE 1
 9132  042260  042356                          WORD    QDRV2           ;DRIVE 2
 9133  042262  042376                          WORD    QDRV3           ;DRIVE 3
 9134  042264  042416                          WORD    QDRV4           ;DRIVE 4
 9135  042266  042436                          WORD    QDRV5           ;DRIVE 5
 9136  042270  042456                          WORD    QDRV6           ;DRIVE 6
 9137  042272  042476                          WORD    QDRV7           ;DRIVE 7
```

```
 9138
 9139  042274  042316                QSTART  .WORD   QDRV0           ;DRIVE 0 START ADDRESS
 9140  042276  042336                QSTOP.  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
 9141  042300  042356                        WORD    QDRV2           ;STOP DRIVE 1--START DRIVE 2
 9142  042302  042376                        WORD    QDRV3           ;STOP DRIVE 2--START DRIVE 3
 9143  042304  042416                        WORD    QDRV4           ,STOP DRIVE 3--START DRIVE 4
 9144  042306  042436                        WORD    QDRV5           ;STOP DRIVE 4--START DRIVE 5
 9145  042310  042456                        WORD    QDRV6           ,STOP DRIVE 5--START DRIVE 6
 9146  042312  042476                        WORD    QDRV7           ,STOP DRIVE 6--START DRIVE 7
 9147  042314  042516                        WORD    QTERM           ,STOP DRIVE 7
 9148
 9149                                ,DRIVE REQUEST QUEUES
 9150
 9151  042316  000010                QDRV0    BLKW   10
 9152  042336  000010                QDRV1   .BLKW   10
 9153  042356  000010                QDRV2    BLKW   10
 9154  042376  000010                QDRV3   .BLKW   10
 9155  042416  000010                QDRV4   .BLKW   10
 9156  042436  000010                QDRV5    BLKW   10
 9157  042456  000010                QDRV6    BLKW   10
 9158  042476  000010                QDRV7    BLKW   10
 9159          042516                QTERM=
 9160
 9161                                ,ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
 9162                                ,
 9163                                ,CALL
 9164                                ,       JSR     PC,CLRQUE
 9165
 9166  042516  104412                CLRQUE  SAVREG                  ,SAVE R0 - R5
 9167  042520  012702  042224                MOV     #QCNT,R2        ,ZERO THE QUEUE COUNTS
 9168  042524  005022                        CLR     (R2)+           ,DRIVES 0 & 1
 9169  042526  005022                        CLR     (R2)+           ,DRIVES 2 & 3
 9170  042530  005022                        CLR     (R2)+           ,DRIVES 4 & 5
 9171  042532  005022                        CLR     (R2)+           ,DRIVES 6 & 7
 9172  042534  012703  000010                MOV     #8.,R3          ,MOVE THE STARTING
 9173  042540  012701  042274                MOV     #QSTART,R1      ,ADDRESS OF THE QUEUE INTO
 9174  042544  012122                1$      MOV     (R1)+,(R2)+     ,THE QUEUE INPUT POINTER
 9175  042546  005303                        DEC     R3
 9176  042550  001375                        BNE     1$
 9177  042552  012703  000010                MOV     #8.,R3          ,MOVE THE STARTING ADDRESS
 9178  042556  012701  042274                MOV     #QSTART,R1      ,OF THE QUEUE INTO THE
 9179  042562  012122                2$      MOV     (R1)+,(R2)+     ,QUEUE OUTPUT POINTER
 9180  042564  005303                        DEC     R3
 9181  042566  001375                        BNE     2$
 9182  042570  104413                        RESREG                  ,RESTORE R0 - R5
 9183  042572  000207                        RTS     PC
 9184
 9185                                ,EMPTY THE QUEUE SPECIFIED BY R1
 9186                                ,
 9187                                ,CALL
 9188                                ,       MOV     DRVNUM,R1       ,DRIVE NUMBER TO R1
 9189                                ,       JSR     PC,EMPTYQ
 9190
 9191  042574  105061  042224        EMPTYQ  CLRB    QCNT(R1)        ,CLEAR NUMBER OF ITEMS IN QUEUE
 9192  042600  006301                        ASL     R1
 9193  042602  016161  042234  042254        MOV     QINPT(R1),QOUTPT(R1)  SET OUTPUT QUEUE POINTER=INPUT POINTER
```

```
 9194  042610  006201                          ASR     R1
 9195  042612  000207                          RTS     PC
 9196
 9197                                  ;ROUTINE TO PUT A REQUEST IN QUEUE
 9198                                  ;
 9199                                  ;CALL
 9200                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
 9201                                  ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
 9202                                  ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
 9203                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
 9204                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
 9205
 9206  042614  122761  000010  042224  DRVQUE  CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
 9207  042622  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
 9208  042624  105261  042224                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
 9209  042630  006301                          ASL     R1
 9210  042632  010271  042234                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
 9211  042636  062761  000002  042234          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
 9212  042644  026161  042234  042276          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
 9213  042652  001003                          BNE     1$              ;BRANCH IF NO
 9214  042654  016161  042274  042234          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
 9215  042662  006201                  1$      ASR     R1
 9216  042664  005720                          TST     (R0)+           ;TAKE RETURN 2
 9217  042666  000200                  2$      RTS     R0              ;RETURN TO USER
 9218
 9219                                  ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
 9220                                  ;
 9221                                  ;CALL
 9222                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 9223                                  ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
 9224                                  ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
 9225                                  ;                               ;R2=0 IF NO REQUEST IN QUEUE
 9226
 9227  042670  005002                  GETREQ  CLR     R2
 9228  042672  105761  042224                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
 9229  042676  001404                          BEQ     2$              ;NO---BRANCH
 9230  042700  006301                  1$      ASL     R1
 9231  042702  017102  042254                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
 9232  042706  006201                          ASR     R1
 9233  042710  000207                  2$      RTS     PC              ;RETURN TO USER
 9234
 9235                                  ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
 9236                                  ;
 9237                                  ;CALL
 9238                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 9239                                  ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
 9240                                  ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
 9241
 9242  042712  105361  042224          POPQUE  DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
 9243  042716  006301                          ASL     R1
 9244  042720  017102  042254                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
 9245  042724  062761  000002  042254          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
 9246  042732  026161  042254  042276          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
 9247  042740  001003                          BNE     1$              ;NO--BRANCH TO EXIT
 9248  042742  016161  042274  042254          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
 9249  042750  006201                  1$      ASR     R1
```

```
 9250  042752  000207                    RTS     PC              ,RETURN TO USER
 9251
 9252                                    ,ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
 9253                                    ,REPORTS AN ERROR DIRECTLY
 9254                                    ,
 9255                                    ,CALL
 9256                                    ,       JSR     R0,ES SAV
 9257                                    ,       ERROR   N               ,,THE ERROR CALL
 9258                                    ,       RETURN                  ,THE RETURN IS PAST THE ERROR CALL
 9259
 9260  042754  012037  042770            ES SAV  MOV     (R0)+,1$        ;GET THE ERROR CALL
 9261  042760  013746  001206                    MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN '$ESCAPE'
 9262  042764  005037  001206                    CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
 9263  042770  000000                    1$       WORD    0               ,THE ERROR CALL IS MOVED HERE
 9264  042772  012637  001206                    MOV     (SP)+,$ESCAPE   ,RESTORE THE ESCAPE ADDRESS
 9265  042776  000200                    RTS     R0              ,RETURN
 9266
 9267
 9268                                     SBTTL  ASCIZ MESSAGES
 9269
 9270  043000  000122                    MSG R.  .ASCIZ  /R/
 9271  043002  041506     000            MSG FC.  ASCIZ  /FC/
 9272  043005     114  000103            MSG LC: .ASCIZ  /LC/
 9273  043010  041506  000047            MSGFCP. .ASCIZ  /FC'/
 9274  043014  041514  000047            MSGLCP. .ASCIZ  /LC'/
 9275  043020  041511     000            MSG IC. .ASCIZ  /IC/
 9276  043023     106  000124            MSG.FT  .ASCIZ  /FT/
 9277  043026  052114     000            MSG LT:  ASCIZ  /LT/
 9278  043031     111  000124            MSG IT. .ASCIZ  /IT/
 9279  043034  051506     000            MSG FS  .ASCIZ  /FS/
 9280  043037     114  000123            MSG.LS  .ASCIZ  /LS/
 9281  043042  040520  000124            MSG PAT.         ASCIZ  /PAT/
 9282  043046  000075                    MSG EQ   ASCIZ  /=/
 9283  043050  005015  047503  052116    MSG.CS   ASCIZ  <CR><LF>/CONTROL SWITCHES=/
 9284  043056  047522  020114  053523
 9285  043064  052111  044103  051505
 9286  043072  000075
 9287
 9288  043074  027440  000040            SLASH   .ASCIZ  @ / @
 9289  043100  047125  052111  051440    SYSTAT   ASCIZ  /UNIT STATUS /<CR><LF><LF>
 9290  043106  040524  052524  035123
 9291  043114  005015  000012
 9292  043120  051104  053111  000105    UNTMSG   ASCIZ  /DRIVE/
 9293  043126  047440  043106  044514    UNTOFF  .ASCIZ  / OFFLINE/
 9294  043134  042516     000
 9295  043137     040  047117  044514    UNTON    ASCIZ  / ONLINE/
 9296  043144  042516     000
 9297  043147     040  047516  020124    NOTPRS   ASCIZ  / NOT PRESENT/
 9298  043154  051120  051505  047105
 9299  043162  000124
 9300  043164  052440  051516  043101    NOTSAF. .ASCIZ  / UNSAFE/
 9301  043172  000105
 9302  043174  047040  052117  051040    NOTRP.   ASCIZ  @ NOT RP04/5/6@
 9303  043202  030120  027464  027465
 9304  043210  000066
 9305  043212  050122  032060     000    RP04B.  .ASCIZ  /RP04/
```

```
 9306  043217     122  030120  000065  RP05     .ASCIZ  /RP05/
 9307  043224  050122  033060     000  RP06      ASCIZ  /RP06/
 9308  043231     015  042012  044522  DRIVES    ASCIZ  <CR><LF>/DRIVE(S) TO BE TESTED /
 9309  043236  042526  051450  020051
 9310  043244  047524  041040  020105
 9311  043252  042524  052123  042105
 9312  043260  000040
 9313  043262  047516  042516     000  NONE      ASCIZ  /NONE/
 9314  043267     054     000          COMMA     ASCIZ  /,/
 9315  043271     015  047012  020117  NOCLOK   .ASCIZ  <CR><LF>/NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
 9316  043276  053513  030461  050055
 9317  043304  041440  047514  045503
 9318  043312  020054  044524  044515
 9319  043320  043516  052040  051505
 9320  043326  051524  053440  046111
 9321  043334  020114  047516  020124
 9322  043342  042502  050040  051105
 9323  043350  047506  046522  042105
 9324  043356     000
 9325  043357     015  005012  042524  TESTNG    ASCIZ  <CR><LF><LF>/TESTING DRIVE /
 9326  043364  052123  047111  020107
 9327  043372  051104  053111  020105
 9328  043400     000
 9329  043401     123  051105  040511  SERIAL    ASCIZ  /SERIAL NUMBER /
 9330  043406  020114  052516  041115
 9331  043414  051105  000040
 9332
 9333  043420  005015  051012  052117  MSG7      ASCIZ  <CR><LF><LF>/ROTATIONAL SPEED TIMES/
 9334  043426  052101  047511  040516
 9335  043434  020114  050123  042505
 9336  043442  020104  044524  042515
 9337  043450  000123
 9338  043452  005015  047412  042516  MSG10A    ASCIZ  <CR><LF><LF>/ONE CYLINDER SEEK TIMES/<CR><LF>/ * FORWARD/
 9339  043460  041440  046131  047111
 9340  043466  042504  020122  042523
 9341  043474  045505  052040  046511
 9342  043502  051505  005015  025040
 9343  043510  043040  051117  040527
 9344  043516  042122     000
 9345  043521     015  020012  020052  MSG10B    ASCIZ  <CR><LF>/ * REVERSE/
 9346  043526  042522  042526  051522
 9347  043534  000105
 9348  043536  005015  040412  041503  MSG11A    ASCIZ  <CR><LF><LF>/ACCESS TIME MEASURMENT/<CR><LF>/ * FORWARD/
 9349  043544  051505  020123  044524
 9350  043552  042515  046440  040505
 9351  043560  052523  046522  047105
 9352  043566  006524  020012  020052
 9353  043574  047506  053522  051101
 9354  043602  000104
 9355  043604  005015  025040  051040  MSG11B   .ASCIZ  <CR><LF>/ * REVERSE/
 9356  043612  053105  051105  042523
 9357  043620     000
 9358  043621     015  005012  040515  MSG12A    ASCIZ  <CR><LF><LF>/MAXIMUM SEEK TIMES/<CR><LF>/ * FORWARD/
 9359  043626  044530  052515  020115
 9360  043634  042523  045505  052040
 9361  043642  046511  051505  005015
```

```
 9362  043650  025040  043040  051117
 9363  043656  040527  042122     000
 9364  043663     015  020012  020052  MSG12B   ASCIZ  <CR><LF>/ * REVERSE/
 9365  043670  042522  042526  051522
 9366  043676  000105
 9367
 9368  043700  005015  044515  036516  MSGMIN:  ASCIZ  <CR><LF>/MIN=/
 9369  043706     000
 9370  043707     015  046412  054101  MSGMAX:  ASCIZ  <CR><LF>/MAX=/
 9371  043714  000075
 9372  043716  005015  053101  036507  MSGAVG:  ASCIZ  <CR><LF>/AVG=/
 9373  043724     000
 9374  043725     060  052440  000123  MSGOUS:  ASCIZ  /0 US/
 9375  043732  041040  046105  053517  MBELOW:  ASCIZ  / BELOW THE MINIMUM OF /
 9376  043740  052040  042510  046440
 9377  043746  047111  046511  046525
 9378  043754  047440  020106     000
 9379  043761     040  041101  053117  MABOVE   ASCIZ  / ABOVE THE MAXIMUM OF /
 9380  043766  020105  044124  020105
 9381  043774  040515  044530  052515
 9382  044002  020115  043117  000040
 9383  044010  051440  042505  051513  MSGNUM   ASCIZ  / SEEKS TIMED/
 9384  044016  052040  046511  042105
 9385  044024     000
 9386  044025     040  047516  020124  MSGNON.  ASCIZ  / NOT TIMED/
 9387  044032  044524  042515  000104
 9388  044040  020040     000          MSG SP   ASCIZ  /  /            ,TWO (2) SPACES
 9389
 9390                                   SBTTL  ERROR HEADER (EM) MESSAGES
 9391
 9392  044043     122  030510  020061  EM1      ASCIZ  /RH11 INTERRUPT OCCURRED (RPAS = 0)/
 9393  044050  047111  042524  051122
 9394  044056  050125  020124  041517
 9395  044064  052503  051122  042105
 9396  044072  024040  050122  051501
 9397  044100  036440  030040  000051
 9398  044106  047125  054105  042520  EM2      ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
 9399  044114  052103  042105  040440
 9400  044122  052124  047105  044524
 9401  044130  047117  047440  041503
 9402  044136  051125  042522  000104
 9403  044144  040515  051523  052502  EM3      ASCIZ  /MASSBUS PARITY ERROR(MCPE=1)/
 9404  044152  020123  040520  044522
 9405  044160  054524  042440  051122
 9406  044166  051117  046450  050103
 9407  044174  036505  024461     000
 9408  044201     115  051501  041123  EM4      ASCIZ  /MASSBUS PARITY ERROR(PAR=1)/
 9409  044206  051525  050040  051101
 9410  044214  052111  020131  051105
 9411  044222  047522  024122  040520
 9412  044230  036522  024461     000
 9413  044235     101  042104  042522  EM5      ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
 9414  044242  051523  050040  052514
 9415  044250  020107  044103  047101
 9416  044256  042507  041040  052111
 9417  044264  051440  052105     000
```

```
 9418  044271     122  030510  027461  EM10    .ASCIZ  "RH11/RP04/5/6 FAILED TO RESPOND TO ADDRESSING"
 9419  044276  050122  032060  032457
 9420  044304  033057  043040  044501
 9421  044312  042514  020104  047524
 9422  044320  051040  051505  047520
 9423  044326  042116  052040  020117
 9424  044334  042101  051104  051505
 9425  044342  044523  043516     000
 9426  044347     104  044522  042526  EM11     ASCIZ  /DRIVE SELECTED IS NOT ONLINE/
 9427  044354  051440  046105  041505
 9428  044362  042524  020104  051511
 9429  044370  047040  052117  047440
 9430  044376  046116  047111  000105
 9431  044404  046511  051120  050117  EM12     ASCIZ  /IMPROPER HEADER DATA/
 9432  044412  051105  044040  040505
 9433  044420  042504  020122  040504
 9434  044426  040524     000
 9435  044431     104  052101  020101  EM13     ASCIZ  /DATA COMPARE FAILURE/
 9436  044436  047503  050115  051101
 9437  044444  020105  040506  046111
 9438  044452  051125  000105
 9439  044456  044504  045523  042440  EM17     ASCIZ  /DISK ERROR IN TIMING TEST/
 9440  044464  051122  051117  044440
 9441  044472  020116  044524  044515
 9442  044500  043516  052040  051505
 9443  044506  000124
 9444  044510  046103  041517  020113  EM20     ASCIZ  /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
 9445  044516  045450  030527  026461
 9446  044524  024520  047440  042526
 9447  044532  043122  047514  020127
 9448  044540  047111  052040  046511
 9449  044546  047111  020107  042524
 9450  044554  052123     000
 9451  044557     104  051511  020113  EM23     ASCIZ  /DISK ERROR DURING SEEK/
 9452  044564  051105  047522  020122
 9453  044572  052504  044522  043516
 9454  044600  051440  042505  000113
 9455  044606  042523  045505  047040  EM24     ASCIZ  /SEEK NOT COMPLETE WITHIN 120 MS/
 9456  044614  052117  041440  046517
 9457  044622  046120  052105  020105
 9458  044630  044527  044124  047111
 9459  044636  030440  030062  046440
 9460  044644  000123
 9461  044646  044122  030461  051057  EM41     ASCIZ  "RH11/RP04/5/6 ERROR"
 9462  044654  030120  027464  027465
 9463  044662  020066  051105  047522
 9464  044670  000122
 9465  044672  040506  040524  020114  EM46     ASCIZ  /FATAL WRITE CHECK ERROR/
 9466  044700  051127  052111  020105
 9467  044706  044103  041505  020113
 9468  044714  051105  047522  000122
 9469
 9470                                  .SBTTL  STATUS/ERROR INDICATOR MESSAGES
 9471
 9472  044722  043117  046106  047111  MSGB14   ASCIZ  /OFFLINE OR UNSAFE DRIVE REQUESTED/
 9473  044730  020105  051117  052440
```

```
 9474  044736  051516  043101  020105
 9475  044744  051104  053111  020105
 9476  044752  042522  052521  051505
 9477  044760  042524  000104
 9478  044764  047125  047514  042101  MSGB13   ASCIZ  /UNLOADED DRIVE REQUESTED/
 9479  044772  042105  042040  044522
 9480  045000  042526  051040  050505
 9481  045006  042525  052123  042105
 9482  045014     000
 9483  045015     120  051105  044523  MSGB12   ASCIZ  /PERSISTENT UNSAFE/
 9484  045022  052123  047105  020124
 9485  045030  047125  040523  042506
 9486  045036     000
 9487  045037     120  051101  052111  MSGB11   ASCIZ  /PARITY ERROR OCCURRED/
 9488  045044  020131  051105  047522
 9489  045052  020122  041517  052503
 9490  045060  051122  042105     000
 9491  045065     106  052101  046101  MSGB10   ASCIZ  /FATAL PARITY ERROR/
 9492  045072  050040  051101  052111
 9493  045100  020131  051105  047522
 9494  045106  000122
 9495  045110  047523  052106  040527  MSGB09   ASCIZ  /SOFTWARE TIMEOUT ON THIS DRIVE/
 9496  045116  042522  052040  046511
 9497  045124  047505  052125  047440
 9498  045132  020116  044124  051511
 9499  045140  042040  044522  042526
 9500  045146     000
 9501  045147     123  043117  053524  MSGB08   ASCIZ  /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
 9502  045154  051101  020105  044524
 9503  045162  042515  052517  020124
 9504  045170  047117  040440  047516
 9505  045176  044124  051105  042040
 9506  045204  044522  042526     000
 9507  045211     105  051122  051117  MSGB06   ASCIZ  "ERROR OCCURRED DUPING I/O OPERATION"
 9508  045216  047440  041503  051125
 9509  045224  042522  020104  052504
 9510  045232  044522  043516  044440
 9511  045240  047457  047440  042520
 9512  045246  040522  044524  047117
 9513  045254     000
 9514  045255     105  051122  051117  MSGB05   ASCIZ  "ERROR OCCURRED DURING NON-I/O OPERATION"
 9515  045262  047440  041503  051125
 9516  045270  042522  020104  052504
 9517  045276  044522  043516  047040
 9518  045304  047117  044455  047457
 9519  045312  047440  042520  040522
 9520  045320  044524  047117     000
 9521  045325     125  051516  043101  MSGB04   ASCIZ  /UNSAFE OCCURRED/
 9522  045332  020105  041517  052503
 9523  045340  051122  042105     000
 9524  045345     101  052125  046517  MSGB03   ASCIZ  /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
 9525  045352  052101  041511  051040
 9526  045360  041505  046101  041111
 9527  045366  040522  042524  051440
 9528  045374  050505  042525  041516
 9529  045402  020105  041517  052503
```

```
 9530  045410  051122  042105     000
 9531  045415     104  044522  042526  MSGB02   ASCIZ  /DRIVE HAS NOT RESPONDED TO PORT REQUEST/
 9532  045422  044040  051501  047040
 9533  045430  052117  051040  051505
 9534  045436  047520  042116  042105
 9535  045444  052040  020117  047520
 9536  045452  052122  051040  050505
 9537  045460  042525  052123     000
 9538  045465     104  044522  042526  MSGB01   ASCIZ  /DRIVE HAS BECOME NON-EXISTENT/
 9539  045472  044040  051501  041040
 9540  045500  041505  046517  020105
 9541  045506  047516  026516  054105
 9542  045514  051511  042524  052116
 9543  045522     000
 9544
 9545                            SBTTL  DATA HEADER (DT) MESSAGES
 9546
 9547  045523     105  051122  050040  DH1      ASCIZ  /ERR PC  RPAS/
 9548  045530  020103  051040  040520
 9549  045536  000123
 9550  045540  051105  020122  041520  DH2      ASCIZ  /ERR PC  DRIVE   RPAS    RPDS1   RPER1   RPER2   RPER3/
 9551  045546  020040  051104  053111
 9552  045554  020105  020040  050122
 9553  045562  051501  020040  020040
 9554  045570  050122  051504  020061
 9555  045576  020040  050122  051105
 9556  045604  020061  020040  050122
 9557  045612  051105  020062  020040
 9558  045620  050122  051105  000063
 9559  045626  042524  052123  020040  DH3      ASCIZ  /TEST    ERR PC  ADDRESS DATA/
 9560  045634  020040  051105  020122
 9561  045642  041520  020040  042101
 9562  045650  051104  051505  020123
 9563  045656  040504  040524     000
 9564  045663     124  051505  020124  DH4      ASCIZ  /TEST    ERR PC  ADDRESS GDDATA  BDDATA/
 9565  045670  020040  042440  051122
 9566  045676  050040  020103  040440
 9567  045704  042104  042522  051523
 9568  045712  043440  042104  052101
 9569  045720  020101  041040  042104
 9570  045726  052101  000101
 9571  045732  050122  051503  020061  DH10     ASCIZ  /RPCS1   ERR PC/
 9572  045740  020040  051105  020122
 9573  045746  041520     000
 9574  045751     104  044522  042526  DH11     ASCIZ  /DRIVE   ERR PC/
 9575  045756  020040  042440  051122
 9576  045764  050040  000103
 9577  045770  042524  052123  020040  DH12     ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR/
 9578  045776  020040  051105  020122
 9579  046004  041520  020040  051524
 9580  046012  020124  041520  020040
 9581  046020  051104  053111  020105
 9582  046026  020040  054503  047114
 9583  046034  051104  020040  051124
 9584  046042  041501  020113  020040
 9585  046050  042523  052103  051117
```

```
 9586  046056     000
 9587  046057     107  041504  046131  DH12A   ASCIZ  /GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR/
 9588  046064  020040  043440  052104
 9589  046072  045522  020040  043440
 9590  046100  051504  052103  020122
 9591  046106  041040  041504  046131
 9592  046114  020040  041040  052104
 9593  046122  045522  020040  041040
 9594  046130  051504  052103  000122
 9595  046136  042107  040504  020124  DH13A   ASCIZ  /GDDAT   BDDAT   WRDCNT  GDADR   BDADR/
 9596  046144  020040  042102  040504
 9597  046152  020124  020040  051127
 9598  046160  041504  052116  020040
 9599  046166  042107  042101  020122
 9600  046174  020040  042102  042101
 9601  046202  000122
 9602  046204  042524  052123  020040  DH17    ASCIZ  /TEST    ERR PC  DRIVE   RPCS1   RPDS1   RPER1   RPER2   RPER3/
 9603  046212  020040  051105  020122
 9604  046220  041520  020040  051104
 9605  046226  053111  020105  020040
 9606  046234  050122  051503  020061
 9607  046242  020040  050122  051504
 9608  046250  020061  020040  050122
 9609  046256  051105  020061  020040
 9610  046264  050122  051105  020062
 9611  046272  020040  050122  051105
 9612  046300  000063
 9613  046302  042524  052123  020040  DH21    ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK/
 9614  046310  020040  051105  020122
 9615  046316  041520  020040  051524
 9616  046324  020124  041520  020040
 9617  046332  051104  053111  020105
 9618  046340  020040  054503  047114
 9619  046346  051104  020040  051124
 9620  046354  041501  000113
 9621  046360  042107  040504  020124  DH21A   ASCIZ  /GDDAT   BDDAT   WRDCNT  SECTOR/
 9622  046366  020040  042102  040504
 9623  046374  020124  020040  051127
 9624  046402  041504  052116  020040
 9625  046410  042523  052103  051117
 9626  046416     000
 9627  046417     124  051505  020124  DH23    ASCIZ  /TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2   RPDS1/
 9628  046424  020040  042440  051122
 9629  046432  050040  020103  042040
 9630  046440  044522  042526  020040
 9631  046446  041440  046131  042116
 9632  046454  020122  051040  041520
 9633  046462  030523  020040  051040
 9634  046470  041520  031123  020040
 9635  046476  051040  042120  030523
 9636  046504     000
 9637  046505     122  042520  030522  DH23A   ASCIZ  /RPER1   RPER2   RPER3   RPCA    PPCC/
 9638  046512  020040  051040  042520
 9639  046520  031122  020040  051040
 9640  046526  042520  031522  020040
 9641  046534  051040  041520  020101
```

```
 9642  046542  020040  051040  041520
 9643  046550  000103
 9644  046552  042524  052123  020040  DH41:   ASCIZ  /TEST   ERR PC  TST PC  DRIVE/
 9645  046560  020040  051105  020122
 9646  046566  041520  020040  051524
 9647  046574  020124  041520  020040
 9648  046602  051104  053111  000105
 9649  046610  042524  052123  020040  DH42    ASCIZ  /TEST   ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1/
 9650  046616  020040  051105  020122
 9651  046624  041520  020040  051524
 9652  046632  020124  041520  020040
 9653  046640  051104  053111  020105
 9654  046646  020040  050122  051503
 9655  046654  020061  020040  050122
 9656  046662  051503  020062  020040
 9657  046670  050122  051504  000061
 9658  046676  050122  051105  020061  DH43A   ASCIZ  /RPER1   RPER2   RPER3/
 9659  046704  020040  050122  051105
 9660  046712  020062  020040  050122
 9661  046720  051105  000063
 9662  046724  050122  051503  020061  DH44A   ASCIZ  /RPCS1   RPCS2   RPDS1   RPCC    RPCA    RPDA/
 9663  046732  020040  050122  051503
 9664  046740  020062  020040  050122
 9665  046746  051504  020061  020040
 9666  046754  050122  041503  020040
 9667  046762  020040  050122  040503
 9668  046770  020040  020040  050122
 9669  046776  040504     000
 9670  047001     122  042520  030522  DH44B   ASCIZ  /RPER1   RPER2   RPEP3/
 9671  047006  020040  051040  042520
 9672  047014  031122  020040  051040
 9673  047022  042520  031522     000
 9674  047027     122  042520  030522  DH45A   ASCIZ  /RPER1   RPER2   RPER3   RPWC    RPBA    RPDB/
 9675  047034  020040  051040  042520
 9676  047042  031122  020040  051040
 9677  047050  042520  031522  020040
 9678  047056  051040  053520  020103
 9679  047064  020040  051040  041120
 9680  047072  020101  020040  051040
 9681  047100  042120  000102
 9682
 9683
 9684                          EVEN
 9685
 9686                          SBTTL  DATA TABLE (DT)
 9687
 9688  047104  001116  001170          DT1     WORD   $ERRPC,$REG3
 9689  047110  001116  001164  001170  DT2     WORD   $ERRPC,$REG1,$REG3,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6
 9690  047116  034346  034350  034352
 9691  047124  034354
 9692  047126  001176  001116  041506  DT3     .WORD  $TMP0,$ERRPC,RD.ADR,RD WRD
 9693  047134  041510
 9694  047136  001176  001116  041732  DT4     WORD   $TMP0,$ERRPC,WRT.ADR,WRT WD,RD WRD
 9695  047144  041730  041510
 9696  047150  001176  001116  001164  DT5     WORD   $TMP0,$ERRPC,$REG1,$REG5,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6
 9697  047156  001174  034346  034350
```

```
 9698  047164  034352  034354
 9699  047170  001366  001116          DT10    .WORD   RH.ADR,$ERRPC
 9700  047174  001166  001116          DT11    .WORD   $REG2,$ERRPC
 9701  047200  001176  001116  001162  DT12    .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC DS
 9702  047206  001254  001270  001274
 9703  047214  001272
 9704  047216  001270  001274  001272  DT12A   .WORD   CYL.DS,TRK.DS,SEC.DS,CYL.RD,TRK.RD,SEC RD
 9705  047224  001262  001264  001266
 9706  047232  001176  001116  001162  DT13.   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS.SEC DS
 9707  047240  001254  001270  001274
 9708  047246  001272
 9709  047250  001124  001126  001172  DT13A   .WORD   $GDDAT,$BDDAT,$REG4,$GDADR,$BDADR
 9710  047256  001120  001122
 9711  047262  001176  001116  001254  DT17    .WORD   $TMPO,$ERRPC,CHKDRV,RP.REG,RP.REG+12,RP REG+14,RP REG+40,RP REG+42
 9712  047270  004204  004216  004220
 9713  047276  004244  004246
 9714  047302  001176  001116  001162  DT21    .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS
 9715  047310  001254  001270  001274
 9716  047316  001164  001126  001172  DT21A   .WORD   $REG1,$BDDAT,$REG4,$REG1
 9717  047324  001164
 9718  047326  001176  001116  001254  DT23    .WORD   $TMPO,$ERRPC,CHKDRV,CYL.DS,RP REG,RP REG+10,RP REG+12
 9719  047334  001270  004204  004214
 9720  047342  004216
 9721  047344  004220  004244  004246  DT23A   .WORD   RP.REG+14,RP REG+40,RP.REG+42,RP.REG+34,RP REG+36
 9722  047352  004240  004242
 9723  047356  001176  001116  001162  DT41    .WORD   $TMPO,$ERRPC,$REGO,CHKDRV
 9724  047364  001254
 9725  047366  001176  001116  001162  DT42    .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,RP REG,RP REG+10,RP REG+12
 9726  047374  001254  004204  004214
 9727  047402  004216
 9728  047404  001176  001116  001162  DT43     WORD   $TMPO,$ERRPC,$REGO,CHKDRV,RP REG,RP REG+10,RP REG+12
 9729  047412  001254  004204  004214
 9730  047420  004216
 9731  047422  004220  004244  004246  DT43A    WORD   RP REG+14,RP.REG+40,RP.REG+42
 9732  047430  001176  001116  001162  DT44     WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL DS,TRK DS,SEC DS
 9733  047436  001254  001270  001274
 9734  047444  001272
 9735  047446  004204  004214  004216  DT44A    WORD   RP.REG,RP.REG+10,RP.REG+12,RP REG+36,RP REG+34,RP REG+06
 9736  047454  004242  004240  004212
 9737  047462  004220  004244  004246  DT44B   .WORD   RP REG+14,RP.REG+40,RP.REG+42
 9738  047470  001176  001116  001162  DT45    .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC DS
 9739  047476  001254  001270  001274
 9740  047504  001272
 9741  047506  004204  004214  004216  DT45A   .WORD   RP.REG,RP.REG+10,RP REG+12,RP REG+36,RP REG+34,RP REG+06
 9742  047514  004242  004240  004212
 9743  047522  004220  004244  004246  DT45B    WORD   RP REG+14,RP REG+40,RP REG+42,RP REG+2,RP REG+4,RP REG+22
 9744  047530  004206  004210  004226
 9745
 9746                                  SBTTL   DATA FORMAT (DF) TABLE
 9747
 9748  047536  000001                  DF1     .WORD   1               ,NUMBER OF DATA HEADERS
 9749  047540     002                          BYTE    2               ,NUMBER OF WORDS IN DATA TABLE
 9750  047541     000                          BYTE    0               ,ALL 3 NUMBERS ARE OCTAL
 9751
 9752  047542  000001                  DF2      WORD   1
 9753  047544     007                          BYTE    7
```

```
 9754  047545     000                        BYTE   0
 9755
 9756  047546  000001              DF3       WORD   1
 9757  047550     004                        BYTE   4
 9758  047551     000                        BYTE   0
 9759
 9760  047552  000001              DF4       WORD   1
 9761  047554     005                        BYTE   5
 9762  047555     000                        BYTE   0
 9763
 9764  047556  000001              DF10    . WORD   1
 9765  047560     002                        BYTE   2
 9766  047561     000                     .  BYTE   0
 9767
 9768  047562  000001              DF11    . WORD   1
 9769  047564     002                        BYTE   2
 9770  047565     000                        BYTE   0
 9771
 9772  047566  000002              DF12    . WORD   2              ;2 DH'S TO BE TYPED
 9773  047570     007                        BYTE   7              ,7 DATA WORDS FOLLOW THE 1ST DH
 9774  047571     160                        BYTE   160            ;WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL
 9775  047572  046057                        WORD   DH12A          ;ADDRESS OF 2ND DH
 9776  047574     006                        BYTE   6              ,6 DATA WORDS FOLLOW THE 2ND DH
 9777  047575     000                        BYTE   0              ;ALL WORDS ARE OCTAL
 9778
 9779  047576  000002              DF13    . WORD   2
 9780  047600     007                        BYTE   7
 9781  047601     160                        BYTE   160
 9782  047602  046136                        WORD   DH13A
 9783  047604     005                        BYTE   5
 9784  047605     004                        BYTE   4              ,WORD 3 IS DECIMAL
 9785
 9786  047606  000000              DF14      WORD   0
 9787  047610     005                        BYTE   5
 9788  047611     004                        BYTE   4              ,WORD 3 IS DECIMAL
 9789
 9790  047612  000001              DF17    . WORD   1
 9791  047614     010                     .  BYTE    D8
 9792  047615     000                        BYTE   0
 9793
 9794  047616  000002              DF21    . WORD   2
 9795  047620     006                        BYTE   6
 9796  047621     060                     .  BYTE   60
 9797  047622  046360                     .  WORD   DH21A
 9798  047624     004                        BYTE   4
 9799  047625     014                        BYTE   14
 9800
 9801  047626  000000              DF22    . WORD   0
 9802  047630     004                     .  BYTE   4
 9803  047631     014                     .  BYTE   14
 9804
 9805  047632  000002              DF23    . WORD   2
 9806  047634     007                        BYTE   7
 9807  047635     010                        BYTE   10             ,WORD 4 IS DECIMAL
 9808  047636  046505                     .  WORD   DH23A
 9809  047640     005                     .  BYTE   5
```

```
 9810  047641  000                     .BYTE   0
 9811
 9812
 9813  047642  000001          DF41:   .WORD   1
 9814  047644  004                     BYTE    4
 9815  047645  000                     .BYTE   0
 9816
 9817  047646  000001          DF42    .WORD   1
 9818  047650  007                     BYTE    7
 9819  047651  000                     .BYTE   0
 9820
 9821  047652  000002          DF43    .WORD   2
 9822  047654  007                     .BYTE   7
 9823  047655  000                     .BYTE   0
 9824  047656  046676                  .WORD   DH43A
 9825  047660  003                     BYTE    3
 9826  047661  000                     .BYTE   0
 9827
 9828  047662  000003          DF44.   .WORD   3
 9829  047664  007                     .BYTE   7
 9830  047665  160                     BYTE    160
 9831  047666  046724                  .WORD   DH44A
 9832  047670  006                     .BYTE   6
 9833  047671  000                     .BYTE   0
 9834  047672  047001                  WORD    DH44B
 9835  047674  003                     .BYTE   3
 9836  047675  000                     .BYTE   0
 9837
 9838  047676  000003          DF45    .WORD   3
 9839  047700  007                     .BYTE   7
 9840  047701  160                     .BYTE   160
 9841  047702  046724                  .WORD   DH44A
 9842  047704  006                     BYTE    6
 9843  047705  000                     BYTE    0
 9844  047706  047027                  WORD    DH45A
 9845  047710  006                     BYTE    6
 9846  047711  000                     BYTE    0
 9847
 9848                                  EVEN
 9849          047712                  BUFFER=.
 9850
 9851  047712  005015  046412  044501  TITLE.  .ASCII  <CR><LF><LF>/MAINDEC-11-DZRJA-B/<CR><LF>
 9852  047720  042116  041505  030455
 9853  047726  026461  055104  045122
 9854  047734  026501  006502     012
 9855  047741     122  030120  027464          ASCIZ  @RP04/5/6 MECHANICAL & READ-WRITE TEST@<CR><LF><LF>
 9856  047746  027465  020066  042515
 9857  047754  044103  047101  041511
 9858  047762  046101  023040  051040
 9859  047770  040505  026504  051127
 9860  047776  052111  020105  042524
 9861  050004  052123  005015  000012
 9862  050012  005015  047524  052040  LOADRV: ASCII  <CR><LF>/TO TEST DRIVE 0 REPLACE THE 'XXDP' PACK ON DRIVE 0/<CR><LF>
 9863  050020  051505  020124  051104
 9864  050026  053111  020105  020060
 9865  050034  042522  046120  041501
```

```
 9866  050042  020105  044124  020105
 9867  050050  054047  042130  023520
 9868  050056  050040  041501  020113
 9869  050064  047117  042040  044522
 9870  050072  042526  030040  005015
 9871  050100  044527  044124  040440          ASCII  /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40, AND RESTART/<CR><LF>
 9872  050106  047516  044124  051105
 9873  050114  050040  041501  026113
 9874  050122  041440  042514  051101
 9875  050130  046440  046505  051117
 9876  050136  020131  047514  040503
 9877  050144  044524  047117  032040
 9878  050152  026060  040440  042116
 9879  050160  051040  051505  040524
 9880  050166  052122  005015
 9881  050172  044124  020105  051120          ASCIZ  /THE PROGRAM/<CR><LF>
 9882  050200  043517  040522  006515
 9883  050206  000012
 9884  050210  005015  054523  052123  NOLOAD  ASCIZ  <CR><LF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER W LL BE OVERWRITTEN/<CR><L
 9885  050216  046505  044040  051501
 9886  050224  030440  045466  046440
 9887  050232  046505  051117  026131
 9888  050240  023440  054130  050104
 9889  050246  020047  047514  042101
 9890  050254  051105  053440  046111
 9891  050262  020114  042502  047440
 9892  050270  042526  053522  044522
 9893  050276  052124  047105  005015
 9894  050304  000012
 9895
 9896                                   EVEN
 9897
 9898                                   SBTTL  ROUTINE TO SIZE MEMORY
 9899
 9900                                   ;;*****************************************************************************
 9901                                   ;*CALL
 9902                                   ;*      JSR     PC,$SIZE
 9903                                   ;*      RETURN
 9904                                   ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
 9905
 9906  050306  010046                   $SIZE   MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
 9907  050310  010146                           MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
 9908  050312  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
 9909  050316  013746  000006                   MOV     @#ERRVEC+2,-(SP)
 9910  050322  010600                           MOV     SP,R0           ;;SAVE THE STACK POINTER
 9911                                   ;;SET THE ERRVEC PS TO THE PRESENT PS
 9912  050324  104400                           TRAP                    ;;PUSH OLD PSW AND PC ON STACK
 9913  050326  012637  000006                   MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
 9914  050332  012737  050352  000004           MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
 9915  050340  012701  020000                   MOV     #20000,R1       ;;FIRST ADDRESS
 9916  050344  005711                   1$      TST     (R1)            ;;TEST THIS ADDRESS
 9917  050346  005721                           TST     (R1)+           ;;STEP TO NEXT ADDRESS
 9918  050350  000775                           BR      1$              ;;TRY ANOTHER
 9919  050352  162701  000002           2$      SUB     #2,R1           ;;DROP BACK
 9920  050356  010006                           MOV     R0,SP           ;;RESTORE THE STACK
 9921  050360  012637  000006                   MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
```

```
 9922  050364  012637  000004                MOV     (SP)+,@#ERRVEC
 9923  050370  010137  050402                MOV     R1,$LSTAD       ;;LAST ADDRESS
 9924  050374  012601                        MOV     (SP)+,R1        ;;RESTORE R1
 9925  050376  012600                        MOV     (SP)+,R0        ;;RESTORE R0
 9926  050400  000207                        RTS     PC
 9927  050402  000000              $LSTAD    WORD    0               ;;CONTAINS THE LAST ADDRESS
 9928
 9929                                  ;;*****************************************************************
 9930                                  ;SBTTL  GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
 9931                                  ;THIS ROUTINE IS USED TO ENSURE THE BUS ADDRESS
 9932                                  ;OF THE RH11/RP04 IS SETUP TO READ THE PROPER VALUE
 9933                                  ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 9934                                  ;REQUIRED
 9935                                  ;NOTE  THIS ROUTINE DESTROYS R0-R4
 9936                                  ;CALL
 9937                                  ;
 9938                                  ;       JSR     PC,@#GETADR
 9939                                  ;       RETURN
 9940
 9941  050404  005737  001226          GETADR  TST     @#BUSADR        ;INPUT FROM TTY REQUESTED?
 9942  050410  001427                          BEQ     7$              ;NO--BRANCH
 9943  050412  005037  001226                  CLR     @#BUSADR        ;YES--CLEAR THE REQUEST FLAG
 9944  050416  012700  001366          1$      MOV     #RH ADR,R0      ;FIRST ADDRESS
 9945  050422  012703  050562                  MOV     #MRPCS1,R3      ;"RPCS1="
 9946  050426  011004                          MOV     (R0),R4         ;PRESENT RPCS1 ADDRESS
 9947  050430  004037  031516                  JSR     R0,@#GETNUM     ;GET NEW RPCS1
 9948  050434  000402                          BR      2$              ;COMMA
 9949  050436  000767                          BP      1$              ;PERIOD
 9950  050440  000412                          BR      5$              ;DOUBLE PERIOD
 9951  050442  010420                  2$      MOV     R4,(R0)+        ;SAVE NEW RPCS1
 9952  050444  012703  050573                  MOV     #MRHVEC,R3      ;"RHVEC="
 9953  050450  011004                          MOV     (R0),R4         ;PRESENT RH11 VECTOR ADDRESS
 9954  050452  004037  031516                  JSR     R0,@#GETNUM     ;GET NEW RHVEC
 9955  050456  000402                          BR      3$              ;COMMA
 9956  050460  000756                          BR      1$              ;PERIOD
 9957  050462  000401                          BR      5$              ;DOUBLE PERIOD
 9958  050464  010420                  3$      MOV     R4,(R0)+        ;SAVE NEW RHVEC
 9959  050466  010410                  5$      MOV     R4,(R0)         ;SAVE INPUT
 9960  050470  013701  000004          7$      MOV     @#ERRVEC,R1     ;SAVE THE ERROR VECTOR
 9961  050474  012737  050530  000004          MOV     #8$,@#ERRVEC    ;SETUP FOR TRAP
 9962  050502  005777  130660                  TST     @RH ADR         ;CHECK FOR RH11/RP04
 9963  050506  010137  000004                  MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 9964  050512  012700  001366                  MOV     #RH.ADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
 9965  050516  012701  034514                  MOV     #RPADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
 9966  050522  012021                          MOV     (R0)+,(R1)+     ;BUS ADDRESS
 9967  050524  012021                          MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
 9968  050526  000207                          RTS     PC              ;RETURN
 9969  050530  010137  000004          8$      MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 9970  050534  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
 9971  050536  104010                          ERROR   10              ;REPORT THE ERROR
 9972  050540  005737  000042                  TST     @#42            ;IS THERE A MONITOR?
 9973  050544  001724                          BEQ     1$              ;NO--GO ASK FOR ADDRESS
 9974  050546  005037  001232                  CLR     @#DRVSEL        ;YES--NO DRIVES SELECTED
 9975  050552  005037  01  ,42                 CLR     @#$EOPCT        ;NO PASSES
 9976  050556  000137  017366                  JMP     @#$EOP          ;GO TO END OF PROGRAM
 9977
```

```
 9978  050562  005015  050122  051503  MRPCS1   ASCIZ  <CR><LF>/RPCS1=/
 9979  050570  036461     000
 9980  050573     015  051012  053110  MRHVEC   ASCIZ  <CR><LF>/RHVEC=/
 9981  050600  041505  000075
 9982          000001                   END
```

```
ABS    = 000200
ACL    = 000040
ACTDRV  034432
ACTSTR  034433
ACU    = 100000
AOE    = 001000
ATA    = 100000
ATABIT  034502
ATO    = 000001
AT1    = 000002
AT2    = 000004
AT3    = 000010
AT4    = 000020
AT5    = 000040
AT6    = 000100
AT7    = 000200
A16    = 000400
A17    = 001000
BAI    = 000010
BITS    001424
BIT0   = 000001
BIT00  = 000001
BIT01  = 000002
BIT02  = 000004
BIT03  = 000010
BIT04  = 000020
BIT05  = 000040
BIT06  = 000100
BIT07  = 000200
BIT08  = 000400
BIT09  = 001000
BIT1   = 000002
BIT10  = 002000
BIT11  = 004000
BIT12  = 010000
BIT13  = 020000
BIT14  = 040000
BIT15  = 100000
BIT2   = 000004
BIT3   = 000010
BIT4   = 000020
BIT5   = 000040
BIT6   = 000100
BIT7   = 000200
BIT8   = 000400
BIT9   = 001000
BPTVEC= 000014
BUFFER= 047712
BUSADR  001226
BYPASS  001252
CALL A  025032
CALL B  025144
CALL C  025334
CHKDRV  001254
C11     036054
C13     036162
C14     036270
C15     036646
C16     036670
C17     036704
C17B    036732
C18     037004
CKSCTR  030010
CKSWR  = 104407
CK CHR  033766
CK DEC  033740
CK DIG  034042
CK NUM  034226
CK OCT  033712
CLKSTA  001244
CLOSE   033632
CLR    = 000040
CLRBUF  027742
CLRQUE  042516
CLSWDS  033532
CNTRLC  001224
COMMA   043267
COUNT   027120
CR     = 000015
CRLF   = 000200
CSF    = 000002
CSU    = 000010
CYL DS  001270
CYL RD  001262
C SWR   001220
DATCMP  030450
DCK    = 100000
DCL    = 000100
DCU    = 000001
DDISP  = 177570
DECSEC  027660
DECSK   007066
DELTA   001350
DE1    = 000040
DFF20  = 000002
DFLT    001664
DF1     047536
DF10    047556
DF11    047562
DF12    047566
DF13    047576
DF14    047606
DF17    047612
DF2     047542
DF21    047616
DF22    047626
DF23    047632
DF3     047546
DF4     047552
DF41    047642
DF42    047646
DF43    047652
DF44    047662
DF45    047676
DH1     045523
DH10    045732
DH11    045751
DH12    045770
DH12A   046057
DH13A   046136
DH17    046204
DH2     045540
DH21    046302
DH21A   046360
DH23    046417
DH23A   046505
DH3     045626
DH4     045663
DH41    046552
DH42    046610
DH43A   046676
DH44A   046724
DH44B   047001
DH45A   047027
DIGB   = 000004
DISPLA  001142
DISPRE  000174
DLT    = 100000
DL64   = 000020
DMD    = 000001
DORTI   027052
DPB A   004104
DPB B   004124
DPB C   004144
DPINT   034406
DPR    = 000400
DPRQS   034416
DRIVES  043231
DRQ    = 004000
DRVACT  034356
DRVCAL  025524
DRVCLR= 000111
DRVCL1  025544
DRVINT  034744
DRVMSK  001256
DRVOK   006122
DRVQUE  042614
DRVSEL  001232
DRVSTA  034366
DRVTYP  034376
DRY    = 000200
DSWR   = 177570
DTADPB  004164
DTE    = 010000
DTSY   = 000200
DTUW    034500
DTO0   = 000001
DTO1   = 000002
DTO2   = 000004
DTO3   = 000010
DTO4   = 000020
DTO5   = 000040
DTO6   = 000100
DTO7   = 000200
DTO8   = 000400
DT1     047104
DT10    047170
DT11    047174
DT12    047200
DT12A   047216
DT13    047232
DT13A   047250
DT17    047262
DT2     047110
DT21    047302
DT21A   047316
DT23    047326
DT23A   047344
DT3     047126
DT4     047136
DT41    047356
DT42    047366
DT43    047404
DT43A   047422
DT44    047430
DT44A   047446
DT44B   047462
DT45    047470
DT45A   047506
DT45B   047522
DT5     047150
DVA    = 004000
ECH    = 000100
ECI    = 004000
EMPTYQ  042574
EMTVEC= 000030
EM1     044043
EM10    044271
EM11    044347
EM12    044404
EM13    044431
EM17    044456
EM2     044106
EM20    044510
EM23    044557
EM24    044606
EM3     044144
EM4     044201
EM41    044646
EM46    044672
EM5     044235
ERINDX  026246
ERR    = 040000
ERRVEC= 000004
ERR CT  001364
ES. SAV 042754
EXIT0   006514
EXIT1   006672
EXIT10  012022
EXIT11  012242
EXIT12  013000
EXIT13  013444
EXIT14  014162
EXIT15  014700
EXIT16  015276
EXIT17  015716
EXIT2   007110
EXIT20  016436
EXIT21  017214
EXIT22  017364
EXIT3   007304
EXIT4   007672
EXIT5   010072
EXIT6   010336
EXIT7   010716
EXT1   = 000001
EXT10  = 000010
EXT2   = 000002
EXT20  = 000020
EXT4   = 000004
EXT40  = 000040
FC      001510
FEN    = 000200
FER    = 000020
FILBUF  027704
FILRAN  030766
FMT22  = 010000
FS      001524
FT      001516
F1     = 000002
F2     = 000004
F3     = 000010
F4     = 000020
F5     = 000040
GETADR  050404
```

```
GETNUM  031516
GETREG= 000141
GETREQ  042670
GETSWR  031424
GO    = 000001
GRV   = 000010
GTSWR = 104406
GTTST1  032012
GTTST2  032076
GTTST3  032440
GTTST4  032442
GTTST5  032552
GTTST6  032612
GT PRM  031654
GT PR1  031656
GT PR2  032006
HCE   = 000200
HCI   = 002000
HCRC  = 000400
HT    = 000011
IAE   = 002000
IC      001514
IE    = 000100
ILF   = 000001
ILR   = 000002
INCCYL  027630
INCSK   007024
INCTRK  027600
IOTVEC= 000020
IR    = 000100
ISR     037376
IT      001522
ITEM41  004546
IXE   = 004000
LA      037240
LACNT   034444
LC      001512
LDCMD   024766
LF    = 000012
LKS     001412
LKV     001406
LOADRV  050012
LODFLT  024352
LODPRM  024610
LOP CK  026206
LPB     001422
LPS     001420
LPTAVL  001230
LP AVL  024032
LS      001526
LST   = 002000
LT      001520
MABOVE  043761
MBELOW  043732
MCLK  = 000002
MCPE  = 020000
MCPEMX  034512
MHS   = 001000
MINX  = 000004
MNDLTA  034526
MOH   = 020000
MOL   = 010000
MPE   = 000400
MRD   = 000020
MRHVEC  050573
MRPCS1  050562
MSE   = 000020
MSGAVG  043716
MSGB01  045465
MSGB02  045415
MSGB03  045345
MSGB04  045325
MSGB05  045255
MSGB06  045211
MSGB08  045147
MSGB09  045110
MSGB10  045065
MSGB11  045037
MSGB12  045015
MSGB13  044764
MSGB14  044722
MSGFCP  043010
MSGLCP  043014
MSGMAX  043707
MSGMIN  043700
MSGNON  044025
MSGNUM  044010
MSG. CS 043050
MSG. EQ 043046
MSG. FC 043002
MSG. FS 043034
MSG. FT 043023
MSG. IC 043020
MSG IT  043031
MSG. LC 043005
MSG. LS 043037
MSG. LT 043026
MSG. PA 043042
MSG. R  043000
MSG. SP 044040
MSGOUS  043725
MSG10A  043452
MSG10B  043521
MSG11A  043536
MSG11B  043604
MSG12A  043621
MSG12B  043663
MSG7    043420
MSTCK = 000010
MWR   = 000040
MXDLTA  034524
MXF   = 001000
MXLACT  034522
MXSTAL  001362
MXWNDW  034530
NBA   = 100000
NC1     001532
NC2     001534
NED   = 010000
NEM   = 004000
NHS   = 002000
NOCLOK  043271
NOLOAD  050210
NONE    043262
NOOP  = 000101
NOTPRS  043147
NOTRP   043174
NOTSAF  043164
OCYL  = 100000
OFFSET= 000115
OFREV = 000200
OF100 = 000004
OF200 = 000010
OF25  = 000001
OF400 = 000020
OF50  = 000002
OF800 = 000040
OPE   = 020000
OPI   = 020000
OPNFLG  001240
OPNPAT  033220
OPNPRM  033016
OPNTST  032644
OPNWDS  033342
OPN CT  032650
OPN. N1 033554
OPN N2  033560
OPN X1  033572
OPN. X2 033576
OPN 1   032654
OPN 2   032676
OPT     035574
OR    = 000200
PACK  = 000123
PAR   = 000010
PAT     001530
PAT PT  003044
PAT0    003104
PAT1    003144
PAT10   003604
PAT11   003644
PAT12   003704
PAT13   003744
PAT14   004004
PAT15   004044
PAT2    003204
PAT3    003244
PAT4    003304
PAT5    003344
PAT6    003404
PAT7    003444
PAT8    003504
PAT9    003544
PGE   = 002000
PGM   = 001000
PIP   = 020000
PIRQ  = 177772
PIRQVE= 000240
PKB     001402
PKC     001404
PKCS    001400
PKV     001374
PLU   = 020000
POPQUE  042712
PRM     001504
PRMLMT  001606
PRMMSG  001636
PRMPT   001536
PRM0    002330
PRM1    002344
PRM10   002544
PRM11   002564
PRM12   002606
PRM13   002622
PRM14   002636
PRM15   002652
PRM16   002666
PRM17   002700
PRM2    002372
PRM20   002712
PRM21   002744
PRM22   002760
PRM3    002414
PRM4    002436
PRM5    002460
PRM6    002502
PRM7    002524
PPN   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSEL  = 002000
PSU   = 000001
PSW   = 177776
PTRN15  002742
PWRVEC= 000024
QCNT    042224
QDRV0   042316
QDRV1   042336
QDRV2   042356
QDRV3   042376
QDRV4   042416
QDRV5   042436
QDRV6   042456
QDRV7   042476
QINPT   042234
QOUTPT  042254
QSTART  042274
QSTOP   042276
QTERM = 042516
RANADR  031242
RANCK   031010
RANPAT  031206
RAW   = 000020
RDCHR = 104410
RDLIN = 104411
RDY   = 000200
RD ADR  041506
RD RP   041462
RD RP1  041504
RD RP2  041630
RD RP3  041634
RD RP4  041640
RD WRD  041510
READ  = 000171
READHD= 000173
READIN= 000121
RECAL = 000107
RELEAS= 000113
RESREG= 104413
RESTAR  006102
RESVEC= 000010
RHVEC   001370
RH ADR  001366
RMR   = 000004
RPADR   034514
RPAS  = 000016
RPBA  = 000004
RPCA  = 000034
RPCC  = 000036
```

```
RPCS1 = 000000
RPCS2 = 000010
RPDA  = 000006
RPDB  = 000022
RPDS1 = 000012
RPDT  = 000026
RPEC1 = 000044
RPEC2 = 000046
RPERRS  034346
RPER1 = 000014
RPER2 = 000040
RPER3 = 000042
RPINIT  034532
RPLA  = 000020
RPMR  = 000024
RPOF  = 000032
RPSN  = 000030
RPT     001506
RPTMR   040734
RPVEC   034516
RPWC  = 000002
RP REG  004204
RP04    035302
RP04B   043212
RP05    043217
RP06    043224
RSTRT1  006046
RSTRT2  006072
RTC   = 000117
R6    =%000006
R7    =%000007
SAVCSW  001222
SAVEFG  034454
SAVREG= 104412
SC      037602
SCTRWC= 177400
SC1   = 000100
SC10  = 001000
SC11    040434
SC12    040524
SC13    040574
SC2   = 000200
SC20  = 002000
SC3     037652
SC4     037656
SC5     037670
SC6     040070
SC6A    040200
SC7     040326
SC8     040404
SEARCH= 000131
SEC DS  001272
SEC RD  001266
SEEK  = 000105
SEEKFG  034456
SEKCNT  001346
SEKTMR  001344
SELDRV= 000145
SERIAL  043401
SETBUF  030360
SETFOR= 000143
SETVEC  005336
SET IE  042152
SKI   = 040000
SLASH   043074
SRCHWT  034430
SRCH00  026670
SRTDRV  005572
SRTINT  005304
SRVCLK  024340
STACK = 001100
STALL   026364
STALL0  001336
STALL1  001354
STALL2  001356
STALL3  001360
START   004672
START1  004636
START2  004660
START3  004626
START4  004650
STATBL  004254
STKLMT= 177774
ST0     041026
ST01    041056
ST02    041254
ST03    041324
ST05    041350
ST06    041356
ST07    041414
ST08    041444
ST09    041454
STRTMR  027054
STRT1A  004642
STRT2A  004664
ST CLK  024074
ST.LCL  024304
ST PCL  024242
SVADR   001340
SVRH11  042034
SVSTAT  001260
SWR     001140
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
SYSTAT  043100
TAP   = 040000
TBITVE= 000014
TD      037442
TDF   = 000040
TESTNG  043357
TEST0   006500
TEST1   006656
TEST10  011126
TEST11  012164
TEST12  012432
TEST13  013126
TEST14  013572
TEST15  014310
TEST16  014776
TEST17  015374
TEST2   007004
TEST20  016122
TEST21  016710
TEST22  017304
TEST3   007222
TEST4   007432
TEST5   010004
TEST6   010204
TEST7   010462
TICKMS  001246
TICKUS  001250
TIMER   034460
TIM DN  001314
TIM PT  001332
TIM UP  001276
TITLE   047712
TKVEC = 000060
TPB     001416
TPS     001414
TPVEC = 000064
TRAPVE= 000034
TRCKWC  001352
TRE   = 040000
TRK DS  001274
TRK. RD 001264
TRK1  = 004000
TRK10 = 040000
TRK2  = 010000
TRK20 = 100000
TRK4  = 020000
TRNSWT  034426
TRTVEC= 000014
TSTNMS  001234
TST0    006354
TST1    006516
TST10   010720
TST10A  011364
TST10B  011756
TST11   012024
TST12   012244
TST13   013002
TST14   013446
TST15   014164
TST16   014702
TST17   015300
TST2    006674
TST20   015720
TST21   016440
TST22   017216
TST3    007112
TST4    007306
TST5    007674
TST6    010074
TST7    010340
TUF   = 000100
TWOMS   026446
TYPDS = 104405
TYPE  = 104401
TYPERR  020054
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
T/PTIM  027252
T10     003014
T11     003024
T12     003034
T7A     002774
T7B     003004
ULDFLG  034434
UNLOAD= 000103
UNS   = 040000
UNTMSG  043120
UNTOFF  043126
UNTON   043137
UPE   = 020000
US1   = 000001
US2   = 000002
US4   = 000004
UWR   = 000010
VERIFY  026526
VUF   = 000002
VU30  = 010000
VV    = 000100
WAO   = 000002
WCE   = 040000
WCEFLG  001334
WCF   = 000040
WCU   = 000001
WLE   = 004000
WRCKD = 000151
WRCKHD= 000153
WRITE = 000161
WRL   = 004000
WRTHD = 000163
WRT AD  041732
WRT RP  041642
WRT R1  041726
WRT R2  042016
WRT R3  042024
WRT R4  042030
WRT R5  042032
WRT WD  041730
WRU   = 000400
WSU   = 000004
$AUTOB  001134
$BDADR  001122
$BDDAT  001126
$BELL   001210
$CHARC  020656
$CKSWR  021632
$CMTAG  001100
$CM1  = 000006
$CM2  = 000014
$CM3  = 000006
$CM4  = 000003
$CNTLC  022544
$CNTLG  022556
$CNTLU  022551
$CRLF   001215
$DBLK   021324
```

```
$DB2D   023254          $GET42  017604          $QUES   001214          $SVLAD  022774          $TPS    001150
$DECVL  023434          $GTSWR  021722          $RAND   023510          $SVPC = 000200          $TRAP   023134
$DIV    023612          $HD   = 000000          $RDCHR  022174          $SWR  = 167000          $TRAP2  023156
$DORGN  017624          $HINUM  023606          $RDLIN  022264          $SWRMK= 000000          $TRP  = 000014
$DTBL   021314          $ICNT   001104          $RDSZ = 000024          $TIMES  001204          $TRPAD  023170
$ENDAD  017614          $INTAG  001135          $REGAD  001160          $TKB    001146          $TSTNM  001102
$ENDCT  017550          $ITEMB  001114          $REG0   001162          $TKCNT  021334          $TTYIN  022520
$ENULL  017630          $LF     001216          $REG1   001164          $TKINT  021344          $TYPDS  021110
$EOP    017366          $LONUM  023610          $REG2   001166          $TKQEN= 021344          $TYPE   020442
$EOPCT  017542          $LPADR  001106          $REG3   001170          $TKQIN  021336          $TYPEC  020612
$ERFLG  001103          $LPERR  001110          $REG4   001172          $TKQOU  021340          $TYPEX  020660
$ERMAX  001115          $LSTAD  050402          $REG5   001174          $TKQSR  021342          $TYPOC  020706
$ERROR  017634          $MNEW   022574          $RESRE  023076          $TKS    001144          $TYPON  020722
$ERRPC  001116          $MSWR   022563          $RTNAD  017626          $TKSRV  021414          $TYPOS  020662
$ERRTB  004306          $MXCNT  023036          $SAVRE  023040          $TMP0   001176          $XTSTR  022620
$ERTTL  001112          $NULL   001154          $SB2D   023220          $TMP1   001200          $$GET4= 000000
$ESCAP  001206          $NWTST= 000001          $SCOPE  022606          $TMP2   001202          $OFILL  021105
$FILLC  001156          $OCNT   021104          $SETUP= 000147          $TN   = 000023                  = 050604
$FILLS  001155          $OMODE  021106          $SIZE   050306          $TNPWR  023364
$GDADR  001120          $OVER   023022          $STUP = 177777          $TPB    001152
$GDDAT  001124          $PASS   001100          $SUPRS  023450          $TPFLG  001157

  ABS   050604      000


 ERRORS DETECTED   0

 DSKW DZRJAB,DSKW DZRJAB/SOL=DSKW SYSMAC SML,DSKM DZRJAB 010,DSKM DZRJAB P11
 RUN-TIME   24 34   9 SECONDS
 RUN-TIME RATIO  452/59=7 5
 CORE USED   52K  (103 PAGES)
```

ACO