

# RP04/05/06

READ-WRITE TEST  
CZRJACO

AH-9182C-MC  
COPYRIGHT © 76-78  
FICHE 1 OF 1

AUG 1978  
**digital**  
MADE IN USA

The main body of the document is a large grid of approximately 20 columns and 20 rows of small, individual data blocks. Each block appears to be a miniature version of a data table or a specific test pattern, containing various alphanumeric characters and symbols. The overall layout is highly structured and repetitive, typical of a data storage or testing artifact from the late 1970s.



.REM @

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

PRODUCT CODE: AC-9180C-MC  
PRODUCT NAME: CZRJACO RPO4/5/6 READ-WRITE TEST  
DATE: AUGUST 1978  
MAINTAINER: DIAGNOSTIC ENGINEERING  
AUTHOR: C. HESS

COPYRIGHT (C) 1976, 1978 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.



CONTENTS

1. ABSTRACT
2. REQUIREMENTS
  - 2.1 EQUIPMENT
  - 2.2 PRELIMINARY PROGRAMS
  - 2.3 MEDIA
3. LOADING PROCEDURE
4. STARTING PROCEDURE
  - 4.1 STARTING ADDRESSES
  - 4.2 OPERATOR ACTION
  - 4.3 PROGRAM ACTION
    - 4.3.1 CONTROL SWITCH SELECTION
    - 4.3.2 RH11 - RH70 ADDRESS SELECTION
    - 4.3.3 DRIVE AND PARAMETER SELECTION
5. OPERATING PROCEDURE
  - 5.1 OPERATIONAL SWITCH SETTINGS
  - 5.2 CONTROL SWITCH SETTINGS
6. ERRORS
  - 6.1 ERROR TYPES
  - 6.2 ERROR RECOVERY
7. RESTRICTIONS
8. MISCELLANEOUS
  - 8.1 EXECUTION TIME
  - 8.2 STACK POINTER
  - 8.3 TIMING TEST (TESTS 12 - 15) PRINTOUTS
  - 8.4 END OF TEST
9. PROGRAM DESCRIPTION
10. PROGRAM LISTING



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 RPO4/5/6 DISK DRIVES

2.2 PRELIMINARY PROGRAMS

RPO4/5/6 DISKLESS CONTROLLER TEST  
PART 1 (MAINDEC-11-DZRJG)  
PART 2 (MAINDEC-11-DZRJH)

RPO4/5/6 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 ASSOCIATED 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 CARRIAGE 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/RPO4 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 RPO4/5 DRIVES AND PARAMETERS 'LC' AND



'FC' ARE USED BY RPO6 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 RPO4/5 AND RPO6 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 RPO4/5'S
"LC"	LAST CYLINDER ADDRESS FOR RPO4/5'S
"FC'"	FIRST CYLINDER ADDRESS FOR RPO6'S
"LC'"	LAST CYLINDER ADDRESS FOR RPO6'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
R=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>           ;RUN ALL SEEK TESTS (OPEN FOR CHANGES) AND
TEST=S/D<CR>          ;THE DATA TEST (WITH DEFAULT PARAMETERS).
TEST 0                ;RUN WITH 10 ITERATIONS
  R=10 / <CR>         ;CHANGE FIRST CYLINDER ADDRESS
  FC=0 / 10. <CR>    ;AND START EXECUTION
```



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

EXAMPLE #6  
-----

```
DRIVE=1<CR>
TEST=S/<CR>
TEST 0
R=10 / 100. <CR>
TEST 1
R=100 / 1000. <CR>
TEST 2
R=1 / 10<CR>
FC=0 / 50<CR>
LC=410 / 51. <CR>
TEST 3
R=1. <CR>
TEST 4
R=1.. <CR>
```

; OPEN THE SEEK TESTS (TESTS 0-10)  
; CHANGE TO 100 ITERATIONS, TO TO THE NEXT TEST  
; CHANGE 'R' TO 1000 ITERATIONS, MOVE TO NEXT TEST  
; CHANGE 'R' TO 10 ITERATIONS, GO TO NEXT PARAMETER  
; CHANGE 'FC' TO 50, GO TO NEXT PARAMETER  
; CHANGE 'LC' TO 51, GO TO THE NEXT TEST  
; MOVE TO NEXT TEST  
; USE TEST 4'S PARAMETERS AND START PROGRAM EXECUTION

EXAMPLE #7  
-----

```
DRIVE=1<CR>
TEST=D/<CR>
TEST 20
R=1 / 1000<CR>
FC=0 / 10<CR>
LC=410 / 10<CR>
FC'=0 / <CR>
LC'=814 / <CR>
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>
```

; SELECT AND OPEN THE DATA TEST  
; DO 1000 ITERATION OF TEST PATTERN  
; #8 ON CYLINDER 10, TRACK 2, SECTOR 4  
; RPO6 PARAMETER  
; RPO6 PARAMETER  
; RUN WITH PATTERN #8

EXAMPLE #8  
-----

```
DRIVE=1<CR>
TEST=D/<CR>
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>
```

; USE THE SAME PARAMETERS AS IN EXAMPLE  
; #7, BUT ALSO SPECIFY A DATA PATTERN (PAT #0).



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 RPO6'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 RPO4/5/6 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/RPO4/5/6 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 RPO4/5 DRIVES AND APPROXIMATELY 16.5 MINUTES TO A PASS WITH RPO6 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 RPO4 VENDOR. THE SEEK TIMES SPECIFIED FOR THE RPO4  
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 MEASUREMENTS  
\* 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 NINETEEN 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 IDENTIFYING ITSELF AND DETERMINING ALL DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH THE LOWEST NUMERICAL 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 PARAMETERS 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 UNTIL "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 RPO4, THE OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 US), THE DRIVE WILL REPORT A 'WRU' ERROR. (RPO5/6'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 PCK 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 EPCH CYLINDER TO ALL OTHER CYLINDERS.

BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER BETWEEN 'FC' AND 'LC' FROM CYLINDER '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 REVERSE SEEK FROM CYLINDER 'LC' TO CYLINDER 0. BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASUREMENT. 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 RPO4/5 OR TO 255 (10) FOR AN RPO6.

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 RPO4/5'S AND TO 814 (10) FOR RPO6'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

STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR FOR THAT SECTOR. THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES "R" DEFAULTS TO 20,000.

- 1) GENERATE A RANDOM ADDRESS
- 2) WRITE A RANDOM PATTERN AT THE ADDRESS GENERATED IN 1.
- 3) GENERATE A RANDOM ADDRESS
- 4) READ THE SECTOR AT THE ADDRESS GENERATED IN 3.
- 5) DO A SOFTWARE CHECK OF THE DATA READ IN 4.
- 6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2
- 7) GENERATE A RANDOM ADDRESS
- 8) READ THE SECTOR AT THE ADDRESS GENERATED IN 7.
- 9) DO A SOFTWARE CHECK OF THE DATA READ IN 8
- 10) DO A WRITE CHECK OF THE DATA WRITTEN IN 2

THE TEST USES THE FOLLOWING PARAMETERS:

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

9.19 TEST 22 - RPO4 ACCESS TIME ADJUSTMENT TEST

THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 136 TO ALLOW THE OPERATOR TO ADJUST THE ACCESS TIME ON AN RPO4 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  
 1427  
 1428  
 1429  
 1430  
 1431  
 1432  
 1433  
 1434  
 1435  
 1436  
 1437  
 1438  
 1439  
 1440  
 1441  
 1442  
 1443  
 1444  
 1445  
 1446  
 1447  
 1448  
 1449  
 1450  
 1451  
 1452  
 1453  
 1454  
 1455  
 1456  
 1457  
 1458  
 1459  
 1460  
 1461  
 1462  
 1463  
 1464  
 1465  
 1466  
 1467  
 1468  
 1469  
 1470  
 1471  
 1472  
 1473  
 1474  
 1475  
 1476  
 1477  
 1478  
 1479

```
.TITLE CZRJAC RPO4/5/6 MECHANICAL AND READ/WRITE TEST
;*COPYRIGHT (C) 1976,1978
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY C. MESS
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC.
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
```

.SBTTL CONTROL SWITCH SETTINGS

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

.SBTTL OPERATIONAL SWITCH SETTINGS

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

.SBTTL TRAP CATCHER

```

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 017616  SENDAD      ;; 1) SET LOC. 46 TO ADDRESS OF SENDAD IN .SEOP
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/RPO4/5/6 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)
    
```



1648	000004	US4=	4	; UNIT SELECT (BIT #2)
1649	000010	BAI=	10	; BUS ADDRESS INCREMENT INHIBIT (BIT #3)
1650		; PAT=	20	; MASSBUS PARITY TEST (BIT #4)
1651	000040	CLR=	40	; CLEAR (BIT #5)
1652	000100	IR=	100	; INPUT READY (BIT #6)
1653	000200	OR=	200	; OUTPUT READY (BIT #7)
1654	000400	NPE=	400	; MASS BUS PARITY ERROR (BIT #8)
1655	001000	MXF=	1000	; MISSED TRANSFER ERROR (BIT #9)
1656	002000	PGE=	2000	; PROGRAM ERROR (BIT #10)
1657	004000	NEM=	4000	; NON EXISTENT MEMORY (BIT #11)
1658	010000	NED=	10000	; NON EXISTENT DRIVE (BIT #12)
1659	020000	UPE=	20000	; UNIBUS PARITY ERROR (BIT #13)
1660	040000	WCE=	40000	; WRITE CHECK ERROR (BIT #14)
1661	100000	DLT=	100000	; DATA LATE (BIT #15)
1662				
1663				; DATA BUFFER REGISTER (RPDB)
1664				; (EACH BIT IS CALLED BY BIT NUMBER)
1665				
1666				
1667				; *****
1668				
1669				. SBTTL RPO4/5/6 REGISTERS
1670				
1671				; *****
1672				
1673				; CONTROL AND STATUS 1 REGISTER. (#00)
1674				
1675	000001	GO=	1	; GO BIT (BIT #0)
1676	000002	F1=	2	; FUNCTION CODE BIT #1
1677	000004	F2=	4	; FUNCTION CODE BIT #2
1678	000010	F3=	10	; FUNCTION CODE BIT #3
1679	000020	F4=	20	; FUNCTION CODE BIT #4
1680	000040	F5=	40	; FUNCTION CODE BIT #5
1681	004000	DVA=	4000	; DEVICE AVAILABLE (BIT #11)
1682				
1683				; DRIVE STATUS REGISTER (RPDS1) (#01)
1684				
1685		; DF5=	1	DRIVE FORWARD 5"/SEC. (BIT #0)
1686	000002	DF20=	2	; DRIVE FORWARD 20"/SEC. (BIT #1)
1687	000004	DIGB=	4	; DRIVE TO INNER GUARD BAND (BIT #2)
1688	000010	GRV=	10	; GO REVERSE (BIT #3)
1689	000020	DL64=	20	; DIFFERENCE LESS THAN 64 (BIT #4)
1690	000040	DE1=	40	; DIFFERENCE EQUALS 1 (BIT #5)
1691	000100	VV=	100	; VOLUME VALID (BIT #6)
1692	000200	DRY=	200	; DRIVE READY (BIT #7)
1693	000400	DPR=	400	; DRIVE PRESENT (BIT #8)
1694	001000	PGM=	1000	; PROGRAMABLE (BIT #9)
1695	002000	LST=	2000	; LAST SECTOR TRANSFERRED (BIT #10)
1696	004000	WRL=	4000	; WRITE LOCK (BIT #11)
1697	010000	MOL=	10000	; MEDIUM ON-LINE (BIT #12)
1698	020000	PIP=	20000	; POSITIONING OPERATION IN PROGRESS (BIT #13)
1699	040000	ERR=	40000	; COMPOSITE ERROR (BIT #14)
1700	100000	ATA=	100000	; ATTENTION ACTIVE (BIT #15)
1701				
1702				; ERROR REGISTER #01 (RPER1) (#02)
1703				

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	; DIAGNOSTIC 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	ATO=	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	; DRIVE 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 (BIT #15)



1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815

000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
  
001000  
002000  
004000  
010000  
020000  
040000  
100000  
  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
001000  
002000  
004000  
010000  
020000  
100000  
  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
001000  
002000  
004000  
020000

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

EXT1= 1 ;EXTENSION 1 (BIT #0)  
 EXT2= 2 ;EXTENSION 2 (BIT #1)  
 EXT4= 4 ;EXTENSION 3 (BIT #2)  
 EXT10= 10 ;EXTENSION 4 (BIT #3)  
 EXT20= 20 ;EXTENSION 5 (BIT #4)  
 EXT40= 40 ;EXTENSION 6 (BIT #5)  
 SC1= 100 ;SECTOR COUNT FIELD 0 (BIT #6)  
 SC2= 200 ;SECTOR COUNT FIELD 1 (BIT #7)  
 ;SC4= 400 ;SECTOR COUNT FIELD 2 (BIT #8)  
 SC10= 1000 ;SECTOR COUNT FIELD 3 (BIT #9)  
 SC20= 2000 ;SECTOR COUNT FIELD 4 (BIT #10)  
 TRK1= 4000 ;TRACK FIELD 1 (BIT #11)  
 TRK2= 10000 ;TRACK FIELD 2 (BIT #12)  
 TRK4= 20000 ;TRACK FIELD 3 (BIT #13)  
 TRK10= 40000 ;TRACK FIELD 4 (BIT #14)  
 TRK20= 100000 ;TRACK FIELD 5 (BIT #15)

;RPO4 ERROR REGISTER #2 (RPER2) (#10)

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

;RPO5/6 ERROR REGISTER #02 (RPER2) (#10)

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

;OFFSET REGISTER (RPOF) (#11)

```
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      ; RPO5/6 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	SELDIV=145
1885		
1886		; OTHER EQUATES
1887		
1888	177400	SCTRWC = -256.
1889	010000	FMT22=10000
1890		

;WORD COUNT FOR SECTOR  
;FORMAT 22 BIT

```

1891      .SBTTL COMMON TAGS
1892
1893      ;; *****
1894      ; THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1895      ; USED IN THE PROGRAM.
1896
1897      =1100
1898      SCMTAG:      .WORD      0      ;; START OF COMMON TAGS
1899      SPASS:      .WORD      0      ;; CONTAINS PASS COUNT
1900      STSTNM:     .BYTE      0      ;; CONTAINS THE TEST NUMBER
1901      SERFLG:     .BYTE      0      ;; CONTAINS ERROR FLAG
1902      SICNT:      .WORD      0      ;; CONTAINS SUBTEST ITERATION COUNT
1903      SLPADR:     .WORD      0      ;; CONTAINS SCOPE LOOP ADDRESS
1904      SLPERR:     .WORD      0      ;; CONTAINS SCOPE RETURN FOR ERRORS
1905      SERTTL:     .WORD      0      ;; CONTAINS TOTAL ERRORS DETECTED
1906      SITEMB:     .BYTE      0      ;; CONTAINS ITEM CONTROL BYTE
1907      SERMAX:     .BYTE      1      ;; CONTAINS MAX. ERRORS PER TEST
1908      SERRPC:     .WORD      0      ;; CONTAINS PC OF LAST ERROR INSTRUCTION
1909      SGADR:      .WORD      0      ;; CONTAINS ADDRESS OF 'GOOD' DATA
1910      SBDADR:     .WORD      0      ;; CONTAINS ADDRESS OF 'BAD' DATA
1911      SGDDAT:     .WORD      0      ;; CONTAINS 'GOOD' DATA
1912      SBDDAT:     .WORD      0      ;; CONTAINS 'BAD' DATA
1913      .WORD      0      ;; RESERVED--NOT TO BE USED
1914      .WORD      0
1915      SAUTOB:     .BYTE      0      ;; AUTOMATIC MODE INDICATOR
1916      SINTAG:     .BYTE      0      ;; INTERRUPT MODE INDICATOR
1917      .WORD      0
1918      SWR:        .WORD      DSWR    ;; ADDRESS OF SWITCH REGISTER
1919      DISPLAY:    .WORD      DDISP   ;; ADDRESS OF DISPLAY REGISTER
1920      $TKS:      177560      ;; TTY KBD STATUS
1921      $TKB:      177562      ;; TTY KBD BUFFER
1922      $TPS:      177564      ;; TTY PRINTER STATUS REG. ADDRESS
1923      $TPB:      177566      ;; TTY PRINTER BUFFER REG. ADDRESS
1924      $NULL:     .BYTE      0      ;; CONTAINS NULL CHARACTER FOR FILLS
1925      $FILLS:    .BYTE      2      ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
1926      $FILLC:   .BYTE      12     ;; INSERT FILL CHARS. AFTER A "LINE FEED"
1927      $TPFLG:   .BYTE      0      ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1928      $REGAD:   .WORD      0      ;; CONTAINS THE ADDRESS FROM
1929      ;; WHICH ($REGO) WAS OBTAINED
1930      $REGO:     .WORD      0      ;; CONTAINS (($REGAD)+0)
1931      $REG1:     .WORD      0      ;; CONTAINS (($REGAD)+2)
1932      $REG2:     .WORD      0      ;; CONTAINS (($REGAD)+4)
1933      $REG3:     .WORD      0      ;; CONTAINS (($REGAD)+6)
1934      $REG4:     .WORD      0      ;; CONTAINS (($REGAD)+10)
1935      $REG5:     .WORD      0      ;; CONTAINS (($REGAD)+12)
1936      $TMP0:     .WORD      0      ;; USER DEFINED
1937      $TMP1:     .WORD      0      ;; USER DEFINED
1938      $TMP2:     .WORD      0      ;; USER DEFINED
1939      $TIMES:    0      ;; MAX. NUMBER OF ITERATIONS
1940      $ESCAPE:   0      ;; ESCAPE ON ERROR ADDRESS
1941      $BELL:     .ASCIZ    <207><377><377> ;; CODE FOR BELL
1942      $QUES:     .ASCII    /?/      ;; QUESTION MARK
1943      $CRLF:     .ASCII    <15>     ;; CARRIAGE RETURN
1944      $LF:       .ASCIZ    <12>     ;; LINE FEED
1945      ;; *****
1946      CR        =      15
    
```

000377



1947		000012		LF	=	12		
1948	001220	000000		C. SWR:	. WORD	0		; CONTROL SWITCHES
1949	001222	000000		SAVCSM:	. 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, AND -1=KW11-L)
1957								; 16 MILLISECONDS PER CLOCK TICK
1958	001246	000020		TICKMS:	. WORD	16.		; 16666 MICROSECONDS PER CLOCK TICK
1959	001250	040432		TICKUS:	. WORD	16666.		
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 SAVED HERE ON AN ERROR
1964								
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		SEKTR:	. WORD	0		; SEEK TIMER (TEST 10)
1988	001346	000000		SEKCN:	. 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 BEFORE GOING TO THE NEXT TEST
1996								; RESERVED
1997	001365	000			. BYTE	0		
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	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

000102

2012						
2013						
2014	001424	000001	;BIT TABLE			
2015	001426	000002	BITS:	. WORD	BIT00	
2016	001430	000004		. WORD	BIT01	
2017	001432	000010		. WORD	BIT02	
2018	001434	000020		. WORD	BIT03	
2019	001436	000040		. WORD	BIT04	
2020	001440	000100		. WORD	BIT05	
2021	001442	000200		. WORD	BIT06	
2022	001444	000400		. WORD	BIT07	
2023	001446	001000		. WORD	BIT08	
2024	001450	002000		. WORD	BIT09	
2025	001452	004000		. WORD	BIT10	
2026	001454	010000		. WORD	BIT11	
2027	001456	020000		. WORD	BIT12	
2028	001460	040000		. WORD	BIT13	
2029	001462	100000		. WORD	BIT14	
2030	001464	000001		. WORD	BIT15	
2031	001466	000002		. WORD	BIT00	
2032	001470	000004		. WORD	BIT01	
2033	001472	000008		. WORD	BIT02	
2034	001474	000016		. WORD	BIT03	
2035	001476	000032		. WORD	BIT04	
2036	001500	000100		. WORD	BIT05	
2037	001502	000200		. WORD	BIT06	
2038				. WORD	BIT07	

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					
; TABLE OF MESSAGE POINTERS					
2091			PRMMSG: . WORD	MSG. R	
2092	001636	043002	. WORD	MSG. FC	
2093	001640	043004	. WORD	MSG. LC	
2094	001642	043007	. WORD	MSGFCP	
2095	001644	043012	. WORD	MSG LCP	
2096	001646	043016	. WORD	MSG. IC	
2097	001650	043022	. WORD	MSG. FT	
2098	001652	043025	. WORD	MSG. LT	
2099	001654	043030	. WORD	MSG. IT	
2100	001656	043033	. WORD	MSG. FS	
2101	001660	043036	. WORD	MSG. LS	
2102	001662	043041	. WORD		
2103					
; STATUS/ERROR INDICATOR MESSAGES POINTER TABLE					
; DEFAULT VALUES OF TEST PARAMETERS					
2104			DFLT: . WORD	1125, 200, 410, 814, 0, 0	; RECAL/SEEK (T0)
2105			. WORD	3377, 100, 0, 128, 0, 256, 0, 0, 0, 0	; SEEK/SEEK (T1)
2106	001664	001125	000310	000632	
2107	001672	001456	000000	000000	
2108	001700	003377	000144	000000	
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 MEASUREMENT 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.,1,1,0,17777 ; 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. ; RPO4 ACCESS TIME ADJUSTMENT TEST (T22)
2156	002322	000210	000000	000377		

; PARAMETER TABLES

; RECAL/SEEK (T0)

PRM0: . WORD 1125  
 . WORD 200.  
 . WORD 410.  
 . WORD 814.  
 . WORD 0  
 . WORD 0

; SEEK/SEEK (T1)

PRM1: . WORD 3377  
 . WORD 100.

2160						
2161	002330	001125				
2162	002332	000310				
2163	002334	000632				
2164	002336	001456				
2165	002340	000000				
2166	002342	000000				
2167						
2168						
2169	002344	003377				
2170	002346	000144				



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				
2182	002372	001177	; INCREMENT SEEK (T2)	
2183	002374	000001	PRM2: . WORD	1177
2184	002376	000000	. WORD	1
2185	002400	000632	. WORD	0
2186	002402	000000	. WORD	410.
2187	002404	001456	. WORD	0
2188	002406	000001	. WORD	814.
2189	002410	000000	. WORD	1
2190	002412	000000	. WORD	0
2191				
2192				
2193	002414	001177	; STEPPING SEEK (T3)	
2194	002416	000001	PRM3: . WORD	1177
2195	002420	000000	. WORD	1
2196	002422	000400	. WORD	0
2197	002424	000000	. WORD	256.
2198	002426	001000	. WORD	0
2199	002430	000001	. WORD	512.
2200	002432	000000	. WORD	1
2201	002434	000000	. WORD	0
2202				
2203				
2204	002436	001177	; OSCILLATING SEEK (T4)	
2205	002440	000001	PRM4: . WORD	1177
2206	002442	000000	. WORD	1
2207	002444	000632	. WORD	0
2208	002446	000000	. WORD	410.
2209	002450	001456	. WORD	0
2210	002452	000001	. WORD	814.
2211	002454	000000	. WORD	1
2212	002456	000000	. WORD	0
2213				
2214				
2215	002460	001177	; CONVERGING/DIVERGING SEEK (T5)	
2216	002462	000001	PRM5: . WORD	1177
2217	002464	000000	. WORD	1
2218	002466	000632	. WORD	0
2219	002470	000000	. WORD	410.
2220	002472	001456	. WORD	0
2221	002474	000001	. WORD	814.
2222	002476	000000	. WORD	1
2223	002500	000000	. WORD	0
2224				
2225				
2226	002502	001177	; SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)	
			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				
2237	002524	000337	;RANDOM SEEK TEST (T7)	
2238	002526	011610	PRM7: . WORD	337
2239	002530	000000	. WORD	5000.
2240	002532	000632	. WORD	0
2241	002534	000000	. WORD	410.
2242	002536	001456	. WORD	0
2243	002540	000000	. WORD	814.
2244	002542	000022	. WORD	0
2245			. WORD	18.
2246				
2247	002544	000177	;SERVO SETTLE DOWN TEST (T10)	
2248	002546	000001	PRM10: . WORD	177
2249	002550	000000	. WORD	1
2250	002552	000632	. WORD	0
2251	002554	000000	. WORD	410.
2252	002556	001456	. WORD	0
2253	002560	000144	. WORD	814.
2254	002562	000000	. WORD	100.
2255			. WORD	0
2256				
2257	002564	001177	;ALL SEEKS TEST (T11)	
2258	002566	000001	PRM11: . WORD	1177
2259	002570	000000	. WORD	1
2260	002572	000632	. WORD	0
2261	002574	000000	. WORD	410.
2262	002576	001456	. WORD	0
2263	002600	000001	. WORD	814.
2264	002602	000000	. WORD	1
2265	002604	000000	. WORD	0
2266			. WORD	0
2267				
2268	002606	001113	;ROTATIONAL SPEED TIMING TEST (T12)	
2269	002610	000001	PRM12: . WORD	1113
2270	002612	000000	. WORD	1
2271	002614	000000	. WORD	0
2272	002616	000000	. WORD	0
2273	002620	000000	. WORD	0
2274			. WORD	0
2275				
2276	002622	000037	;ONE CYLINDER SEEK TIMING TEST (T13)	
2277	002624	000001	PRM13: . WORD	37
2278	002626	000000	. WORD	1
2279	002630	000632	. WORD	0
2280	002632	000000	. WORD	410.
2281	002634	001456	. WORD	0
2282			. WORD	814.



```
2283 ;ACCESS TIME MEASUREMENT 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 ;RPO4 ACCESS TIME ADJUSTMENT TEST (T22)
2337 002760 000037 PRM22: .WORD 37
2338 002762 011610 .WORD 5000.
```

2339 002764 000000  
 2340 002766 000210  
 2341 002770 000000  
 2342 002772 000377  
 2343  
 2344  
 2345  
 2346  
 2347  
 2348 002774 043422  
 2349 002776 000000  
 2350 003000 003142  
 2351 003002 003244  
 2352  
 2353 003004 043422  
 2354 003006 000000  
 2355 003010 003131  
 2356 003012 003255  
 2357  
 2358 003014 043454  
 2359 003016 043523  
 2360 003020 000000  
 2361 003022 001750  
 2362  
 2363 003024 043540  
 2364 003026 043606  
 2365 003030 000000  
 2366 003032 012574  
 2367  
 2368 003034 043623  
 2369 003036 043665  
 2370 003040 000000  
 2371 003042 012574  
 2372  
 2373 003044 003104  
 2374 003046 003144  
 2375 003050 003204  
 2376 003052 003244  
 2377 003054 003304  
 2378 003056 003344  
 2379 003060 003404  
 2380 003062 003444  
 2381 003064 003504  
 2382 003066 003544  
 2383 003070 003604  
 2384 003072 003644  
 2385 003074 003704  
 2386 003076 003744  
 2387 003100 004004  
 2388 003102 004044  
 2389  
 2390  
 2391  
 2392 003104 165555  
 2393 003106 133333  
 2394 003110 165555

```

        . WORD 0
        . WORD 136.
        . WORD 0
        . WORD 255.

;SEEK TIMING LIMITS
T7A:   . WORD MSG7
        . WORD 0
        . WORD 1634.      ; (16.67-2%)*2
        . WORD 1700.      ; (16.67+2%)*2

T7B:   . WORD MSG7
        . WORD 0
        . WORD 1625.      ; (16.67-2.5%)*2
        . WORD 1709.      ; (16.67+2.5%)*2

T10:   . WORD MSG10A
        . WORD MSG10B
        . WORD 0           ; NO LOWER LIMIT
        . WORD 1000.      ; (7+3)*2

T11:   . WORD MSG11A
        . WORD MSG11B
        . WORD 0           ; NO LOWER LIMIT
        . WORD 5500.      ; (28+2)*2

T12:   . WORD MSG12A
        . WORD MSG12B
        . WORD 0           ; NO LOWER LIMIT
        . WORD 5500.      ; (50+2)*2

PAT. PT: . WORD PAT0
        . WORD PAT1
        . WORD PAT2
        . WORD PAT3
        . WORD PAT4
        . WORD PAT5
        . WORD PAT6
        . WORD PAT7
        . WORD PAT8
        . WORD PAT9
        . WORD PAT10
        . WORD PAT11
        . WORD PAT12
        . WORD PAT13
        . WORD PAT14
        . WORD PAT15

; PATTERNS 0 THRU 15
PATO:  . WORD 165555      ; PATTERN 0
        . WORD 133333
        . 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	

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

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 ; DPB (DATA PARAMETER BLOCK)

2664					
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	047714	. 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/RPO4
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	047714	. 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/RPO4
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	047714	. 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/RPO4
2730					; REGISTERS ON AN ERROR. IF LEFT



2731					; ZERO REGISTERS ARE NOT SAVED.	
2732	004162	000000	. WORD	0	; (16) STATUS/ERROR INDICATOR	
2733					; BIT15=1=>ERROR OCCURRED	
2734					; BIT07=1=>DONE	
2735					; BIT14-BIT09 AND BIT06-BIT03	
2736					; INDICATE TYPE OF ERROR	
2737						
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	047714		. 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/RPO4
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/RPO4 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	; RPOA (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	; RPOB (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						

			; STATUS/ERROR MESSAGE POINTER TABLE	
2787				
2788	004254	044724	STATBL: .WORD	MSGB14 ; OFFLINE OR UNSAFE DRIVE REQUESTED
2789	004256	044766	.WORD	MSGB13 ; UNLOAD DRIVE REQUESTED
2790	004260	045017	.WORD	MSGB12 ; PERSISTENT UNSAFE
2791	004262	045041	.WORD	MSGB11 ; PARITY ERROR OCCURRED
2792	004264	045067	.WORD	MSGB10 ; FATAL PARITY ERROR
2793	004266	045112	.WORD	MSGB09 ; SOFTWARE TIMEOUT ON THIS DRIVE
2794	004270	045151	.WORD	MSGB08 ; SOFTWARE TIMEOUT ON ANOTHER DRIVE
2795	004272	045213	.WORD	MSGB06 ; ERROR OCCURRED DURING I/O OPERATION
2796	004274	045257	.WORD	MSGB05 ; ERROR OCCURRED DURING NON-I/O OPERATION
2797	004276	045327	.WORD	MSGB04 ; UNSAFE OCCURRED
2798	004300	045347	.WORD	MSGB03 ; AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
2799	004302	045417	.WORD	MSGB02 ; DRIVE HAS NOT RESPONDED TO PORT REQUEST
2800	004304	045467	.WORD	MSGB01 ; DRIVE HAS BECOME NONEXISTENT



2801  
2802  
2803  
2804  
2805  
2806  
2807  
2808  
2809  
2810  
2811  
2812  
2813  
2814  
2815  
2816  
2817  
2818  
2819  
2820  
2821  
2822  
2823  
2824  
2825  
2826  
2827  
2828  
2829  
2830  
2831  
2832  
2833  
2834  
2835  
2836  
2837  
2838  
2839  
2840  
2841  
2842  
2843  
2844  
2845  
2846  
2847  
2848  
2849  
2850  
2851  
2852  
2853  
2854  
2855  
2856

004306

004306 044045  
004310 045525  
004312 047106  
004314 047540

004316 044110  
004320 045542  
004322 047112  
004324 047544

004326 044146  
004330 045630  
004332 047130  
004334 047550

.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

\$ERRTB:  
;\* EM AND DH ARE ASCII 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

EM1  
DH1  
DT1  
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

EM2  
DH2  
DT2  
DF2

;\* ERROR ITEM 3  
;\* MASSBUS PARITY ERROR (MCPE=1)  
;\* TEST ERR PC ADDRESS DATA  
;\* \$TMPO \$ERRPC RD.ADR RD.WRD

EM3  
DH3  
DT3  
DF3

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

Address	STMP0	SERRPC	WRT. ADR	WRT. WD	RD. WRD
2857					
2858					
2859	004336	044203			
2860	004340	045665			
2861	004342	047140			
2862	004344	047554			
2863					
2864					
2865					
2866					
2867					
2868					
2869	004346	044237			
2870	004350	045542			
2871	004352	047112			
2872	004354	047544			
2873					
2874					
2875					
2876	004356	000000			
2877	004360	000000			
2878	004362	000000			
2879	004364	000000			
2880					
2881					
2882					
2883	004366	000000			
2884	004370	000000			
2885	004372	000000			
2886	004374	000000			
2887					
2888					
2889					
2890					
2891					
2892					
2893	004376	044273			
2894	004400	045734			
2895	004402	047172			
2896	004404	047560			
2897					
2898					
2899					
2900					
2901					
2902					
2903	004406	044351			
2904	004410	045753			
2905	004412	047176			
2906	004414	047564			
2907					
2908					
2909					
2910					
2911					
2912					

```

;* STMP0 SERRPC WRT. ADR WRT. WD RD. WRD
EM4
DH4
DT4
DF4

;* ERROR ITEM 5
;* ADDRESS PLUG CHANGE BIT SET
;* ERR PC DRIVE RPAS RPDS1 RPER1 RPER2 RPER3
;* SERRPC $REG1 $REG3 RPERRS RPERRS+2 RPERRS+4 RPERRS+6
EM5
DH2
DT2
DF2

;* ERROR ITEM 6 -- NOT USED
0
0
0
0

;* ERROR ITEM 7 -- NOT USED
0
0
0
0

;* ERROR ITEM 10
;* RH11/RPO4/5/6 FAILED TO RESPOND TO ADDRESSING
;* RPCS1 ERR PC
;* RH. ADR SERRPC
EM10
DH10
DT10
DF10

;* ERROR ITEM 11
;* DRIVE SELECTED IS NOT ONLINE
;* DRIVE ERR PC
;* $REG2 SERRPC
EM11
DH11
DT11
DF11

;* ERROR ITEM 12
;* IMPROPER HEADER DATA
;* TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
;* STMP0 SERRPC $REG0 CHKDRV CYL. DS TRK. DS SEC. DS
;* GDCYL GDTRK GDSCTR BDCYL BDBTRK BDSCTR
    
```



Line	Code	Value	Description
2913			;* CYL. DS TRK. DS SEC. DS CYL. RD TRK. RD SEC. RD
2914			;* CYLNR, TRACK, AND SECTOR ARE DECIMAL
2915			
2916	004416	044406	EM12
2917	004420	045772	DH12
2918	004422	047202	DT12
2919	004424	047570	DF12
2920			
2921			;* ERROR ITEM 13
2922			;* DATA COMPARE FAILURE
2923			;* TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
2924			;* \$TMPO \$ERRPC \$REGO \$CHKDRV CYL. DS TRK. DS SEC. DS
2925			;* \$GDAT \$BDDAT \$WRDCNT \$GDADR \$BDADR
2926			;* \$SGDAT \$SBDDAT \$REG4 \$SGDADR \$SBDADR
2927			;* CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
2928			
2929	004426	044433	EM13
2930	004430	045772	DH12
2931	004432	047234	DT13
2932	004434	047600	DF13
2933			
2934			;* ERROR ITEM 14 -- FOLLOWS #13
2935			;* \$SGDAT \$SBDDAT \$REG4 \$SGDADR \$SBDADR
2936			
2937	004436	000000	0
2938	004440	000000	0
2939	004442	047252	DT13A
2940	004444	047610	DF14
2941			
2942			;* ERROR ITEM 15
2943			;* DATA COMPARE FAILURE
2944			;* TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR
2945			;* \$TMPO \$ERRPC \$REGO \$CHKDRV CYL. DS TRK. DS SEC. DS
2946			;* \$GDAT \$BDDAT \$WRDCNT \$GDADR \$BDADR
2947			;* \$SGDAT \$SBDDAT \$REG4 \$SGDADR \$SBDADR
2948			;* CYLNR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
2949			
2950	004446	044433	EM13
2951	004450	045772	DH12
2952	004452	047234	DT13
2953	004454	047600	DF13
2954			
2955			;* ERROR ITEM 16 -- FOLLOWS #15
2956			;* \$SGDAT \$SBDDAT \$REG4 \$SGDADR \$SBDADR
2957			
2958	004456	000000	0
2959	004460	000000	0
2960	004462	047252	DT13A
2961	004464	047610	DF14
2962			
2963			;* ERROR ITEM 17
2964			;* DISK ERROR IN TIMING TEST
2965			;* TEST ERR PC DRIVE RPCS1 RPDS1 RPER1 RPER2 RPER3
2966			;* \$TMPO \$ERRPC \$CHKDRV RP. REG RP. REG+12 RP. REG+14 RP. REG+40 RP. REG+42
2967			
2968	004466	044460	EM17

2969	004470	046206	DH17
2970	004472	047264	DT17
2971	004474	047614	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			;* STMPO \$ERRPC CHKDRV RP.REG RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42
2977			
2978	004476	044512	EM20
2979	004500	046206	DH17
2980	004502	047264	DT17
2981	004504	047614	DF17
2982			
2983			;* ERROR ITEM 21
2984			;* DATA COMPARE FAILURE
2985			;* TEST ERR PC TST PC DRIVE CYLNDR TRACK
2986			;* STMPO \$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	044433	EM13
2992	004510	046304	DH21
2993	004512	047304	DT21
2994	004514	047620	DF21
2995			
2996			;* ERROR ITEM 22--FOLLOWS #21
2997			;* \$REG1 \$BDDAT \$REG4 \$REG1
2998			
2999	004516	000000	0
3000	004520	000000	0
3001	004522	047320	DT21A
3002	004524	047630	DF22
3003			
3004			;* ERROR ITEM 23
3005			;* DISK ERROR DURING SEEK
3006			;* TEST ERR PC DRIVE CYLNDR RPCS1 RPCS2 RPDS1
3007			;* STMPO \$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	044561	EM23
3012	004530	046421	DH23
3013	004532	047330	DT23
3014	004534	047634	DF23
3015			
3016			;* ERROR ITEM 24
3017			;* SEEK NOT COMPLETE WITHIN 120 MS
3018			;* TEST ERR PC DRIVE CYLNDR RPCS1 RPCS2 RPDS1
3019			;* STMPO \$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	044610	EM24
3024	004540	046421	DH23



3025 004542 047330  
 3026 004544 047634  
 3027  
 3028  
 3029  
 3030  
 3031  
 3032  
 3033  
 3034  
 3035  
 3036  
 3037  
 3038  
 3039  
 3040  
 3041  
 3042  
 3043  
 3044  
 3045  
 3046  
 3047  
 3048  
 3049 004546  
 3050  
 3051  
 3052  
 3053  
 3054  
 3055  
 3056 004546 044650  
 3057 004550 046554  
 3058 004552 047360  
 3059 004554 047644  
 3060  
 3061  
 3062  
 3063  
 3064  
 3065  
 3066 004556 044650  
 3067 004560 046612  
 3068 004562 047370  
 3069 004564 047650  
 3070  
 3071  
 3072  
 3073  
 3074  
 3075  
 3076  
 3077  
 3078 004566 044650  
 3079 004570 046612  
 3080 004572 047406

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/RPO4/5/6 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
    
```

ITEM41:

```

;* ERROR ITEM 41
;* RH11/RPO4/5/6 ERROR (MESSAGE)
;* TEST ERR PC TST PC DRIVE
;* $TMPO $ERRPC $REGO CHKDRV
    
```

EM41  
 DH41  
 DT41  
 DF41

```

;* ERROR ITEM 42
;* RH11/RPO4/5/6 ERROR (MESSAGE)
;* TEST ERR PC TST PC DRIVE RPCS1 RPCS2 RPDS1
;* $TMPO $ERRPC $REGO CHKDRV RP. REG RP. REG+10 RP. REG+12
    
```

EM41  
 DH42  
 DT42  
 DF42

```

;* ERROR ITEM 43
;* RH11/RPO4/5/6 ERROR (MESSAGE)
;* TEST ERR PC TST PC DRIVE RPCS1 RPCS2 RPDS1
;* $TMPO $ERRPC $REGO CHKDRV RP. REG RP. REG+10 RP. REG+12
;* RPER1 RPER2 RPER3
;* RP. REG+14 RP. REG+40 RP. REG+42
    
```

EM41  
 DH42  
 DT43

3081	004574	047654	DF43
3082			
3083			:* ERROR ITEM 44
3084			:* RH11/RPO4/5/6 ERROR (MESSAGE)
3085			:* TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
3086			:* STMPO \$ERRPC \$REGO 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	044650	EM41
3094	004600	045772	DH12
3095	004602	047432	DT44
3096	004604	047664	DF44
3097			
3098			:* ERROR ITEM 45
3099			:* RH11/RPO4/5/6 ERROR (MESSAGE)
3100			:* TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
3101			:* STMPO \$ERRPC \$REGO 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	044650	EM41
3109	004610	045772	DH12
3110	004612	047472	DT45
3111	004614	047700	DF45
3112			
3113			:* ERROR ITEM 46
3114			:* FATAL WRITE CHECK ERROR (MESSAGE)
3115			:* TEST ERR PC TST PC DRIVE CYLNDR TRACK SECTOR
3116			:* STMPO \$ERRPC \$REGO 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	044674	EM46
3124	004620	045772	DH12
3125	004622	047472	DT45
3126	004624	047700	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 022610 000020          MOV    %SCOPE, @IOTVEC ; IOT VECTOR FOR SCOPE ROUTINE
3151 004722 012737 000340 000022          MOV    #340, @IOTVEC+2 ; LEVEL 7
3152 004730 012737 017636 000030          MOV    %ERROR, @EMTVEC ; EMT VECTOR FOR ERROR ROUTINE
3153 004736 012737 000340 000032          MOV    #340, @EMTVEC+2 ; LEVEL 7
3154 004744 012737 023136 000034          MOV    %STRAP, @TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
3155 004752 012737 000340 000036          MOV    #340, @TRAPVEC+2 ; LEVEL 7
3156 004760 012737 176543 023610          MOV    #176543, %SHNUM ; PRIME THE RANDOM NUMBER GENERATOR
3157 004766 012737 123456 023612          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, %SERMAX ; ALLOW ONE ERROR PER TEST
3161 005012 012737 005012 001106          MOV    #, %SLPADR ; INITIALIZE THE LOOP ADDRESS FOR SCOPE
3162 005020 012737 005020 001110          MOV    #, %SLPERR ; 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    #645, @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    665 ; BRANCH IF NO TIMEOUT TRAP OCCURRED
3171          ; AND THE HARDWARE SWR IS NOT = -1
3172 005064 000403          BR      655 ; BRANCH IF NO TIMEOUT
3173 005066 012716 005074          645: MOV    #655, (SP) ; SET UP FOR TRAP RETURN
3174 005072 000002          RTI
3175 005074 012737 000176 001140          655: MOV    %SWREG, %SWR ; POINT TO SOFTWARE SWR
3176 005102 012737 000174 001142          MOV    %DISPREG, %DISPLAY
3177 005110 012637 000004          665: MOV    (SP)+, @ERRVEC ; RESTORE ERROR VECTOR
3178
3179 005114 012700 001160          MOV    %$REGAD, R0 ; FIRST ADDRESS
3180 005120 005020          15: CLR    (R0)+ ; CLEAR VARIABLE STORAGE
3181 005122 022700 001210          CMP    %$BELL, R0 ; DONE?
3182 005126 001374          BNE    15 ; NO--BRANCH
3183 005130 013737 001414 001150          MOV    @TPS, @STPS ; SETUP THE STATUS AND BUFFER REG'S
    
```

```

3184 005136 013737 001416 001152      MOV      @#TPB, @#STPB      ;FOR THE TYPE ROUTINE
3185 005144 005227 177777                INC      #-1              ;FIRST START ?
3186 005150 001032                BNE     35                ;BR IF NOT
3187 005152 023737 000042 000046      CMP     @#42, @#46       ;ACT11 AUTOMATIC MODE?
3188 005160 001002                BNE     45                ;YES, SKIP TITLE PRINTOUT
3189 005162 104401 047714                TYPE    ,TITLE           ;TYPE THE PROGRAM'S TITLE
3190 005166 005737 000042          45:    TST     42            ;AUTO ACCEPT OR CHAIN MODE ?
3191 005172 001006                BNE     25                ;BR IF EITHER
3192 005174 122737 000011 000041      CMPB   #11, 41          ;LOADED FROM AN RPO4/5/6 ?
3193 005202 001002                BNE     25                ;BR IF NOT
3194 005204 104401 050000                TYPE    ,LOADRV         ;INSTRUCT THE OPERATOR TO REMOVE THE PACK
3195                                ;ON DRIVE 0 IF DRIVE 0 IS TO BE TESTED
3196 005210 105737 000041          25:    TSTB   @#41          ;LOADED FROM PAPER TAPE ?
3197 005214 001410                BEQ     35                ;BR IF NOT
3198 005216 004737 050274                JSR     PC, $SIZE        ;SIZE THE MEMORY
3199 005222 023727 050370 100000      CMP     $LSTAD, #100000 ;16K OR MORE ON THE SYSTEM ?
3200 005230 103002                BHS     35                ;BR IF YES
3201 005232 104401 050176                TYPE    ,NOLOAD         ;INFORM THE OPEATOR THAT THE 'XXDP' LOADER
3202                                ;WILL BE OVERWRITTEN
3203 005236 004737 021346          35:    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     675              ;; BRANCH IF YES
3207 005250 023727 001140 000176      CMP     SWR, #SWREG     ;; SOFTWARE SWITCH REG SELECTED?
3208 005256 001005                BNE     685              ;; BRANCH IF NO
3209 005260 104406                GTSWR                    ;; GET SOFT-SWR SETTINGS
3210 005262 000403                BR      685
3211 005264 112737 000001 001134      675:   MOVB   #1, $AUTOB     ;; SET AUTO-MODE INDICATOR
3212 005272                                685:
3213 005272 005227 177777                INC     #-1              ;SEE IF FIRST START
3214 005276 001002                BNE     SRTINT           ;BR IF NOT
3215 005300 004737 050372                JSR     PC, GETADR       ;GET OR CHECK THE RH11 ADDRESS
3216 005304 104401 001215          SRTINT: TYPE    , $CRLF    ;CR-LF
3217 005310 004737 024034                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 031426                JSR     PC, @#GETSWR     ;GO CHECK FOR CONTROL SWITCHES
3221 005332 004737 024076                JSR     PC, @#ST, CLK    ;INITIALIZE THE CLOCK
3222 005336 004737 034534          SETVEC: JSR     PC, RPINIT     ;CHECK THE DRIVE STATUS
3223 005342 012737 177777 034456      MOV     #-1, $SAVEFG     ;SET THE SAVE REGISTERS FLAG
3224 005350 062727 177777 000000      ADD     #-1, #0          ;FIRST START ?
3225 005356 103003                BCC     115              ;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      115:   MOV     #PR7, PS        ;SET PRIORITY TO ?
3229 005374 005004                CLR     R4              ;DRIVE TABLE POINTER
3230 005376 104401 001215                TYPE    , $CRLF         ;CR-LF
3231 005402 104401 043102                TYPE    , $SYSTAT       ;TYPE STATUS HEADING
3232                                15:
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 044042                TYPE    , MSG, SP       ;SPACES
3239 005420 104401 044042                TYPE    , MSG, SP       ;SPACES

```



3240	005424	105764	034370		TSTB	DRVSTA(R4)		;CHECK DRIVE'S STATUS
3241	005430	100416			BMI	45		;BR IF UNSAFE
3242	005432	001020			BNE	55		;BR IF ONLINE
3243	005434	105764	034400		TSTB	DRVTYP(R4)		;SEE IF OFFLINE OR NONEXISTENT
3244	005440	001404			BEQ	25		;BR IF NONEXISTENT
3245	005442	100006			BPL	35		;BR IF OFFLINE
3246	005444	104401	043176		TYPE	,NOTRP		;DRIVE NOT AN RPO4/5/6
3247	005450	000440			BR	95		;CHECK NEXT DRIVE
3248	005452	104401	043151	25:	TYPE	,NOTPRS		;DRIVE NOT PRESENT
3249	005456	000435			BR	95		;CHECK NEXT DRIVE
3250	005460	104401	043130	35:	TYPE	,UNTOFF		;DRIVE OFFLINE
3251	005464	000405			BR	65		;PRINT DRIVE TYPE
3252	005466	104401	043166	45:	TYPE	,NOTSAF		;DRIVE UNSAFE
3253	005472	000402			BR	65		;PRINT DRIVE TYPE
3254	005474	104401	043141	55:	TYPE	,UNTON		;DRIVE ONLINE
3255	005500	104401	044042	65:	TYPE	,MSG.SP		;SPACES
3256	005504	012737	043214	005550	MOV	#RPO4B,85		;ADDRESS OF RPO4 MESSAGE
3257	005512	132764	000001	034400	BITB	#BIT00,DRVTYP(R4)		;RPO4 ?
3258	005520	001012			BNE	75		;BR IF YES
3259	005522	012737	043221	005550	MOV	#RPO5,85		;ADDRESS OF RPO5 MESSAGE
3260	005530	132764	000002	034400	BITB	#BIT01,DRVTYP(R4)		;RPO5 ?
3261	005536	001003			BNE	75		;BR IF YES
3262	005540	012737	043226	005550	MOV	#RPO6,85		;ADDRESS OF RPO6 MESSAGE
3263	005546	104401			75:	TYPE		;TYPE THE DRIVE TYPE MESSAGE
3264	005550	000000			85:	.WORD	0	;MESSAGE ADDRESS HERE
3265	005552	104401	001215		95:	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	15		;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	15		;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	95		;BR IF NOT
3276	005616	013737	001220	001222	MOV	C.SWR,SAVCSW		;STORE PRESENT 'C.SWR' VALUE
3277	005624	004737	024354		JSR	PC,LODFLT		;RESET PARAMETERS TO THEIR DEFAULT VALUES
3278	005630	004737	031656	95:	JSR	PC,@GT.PRM		;GET PARAMETERS
3279	005634	000420			BR	45		
3280	005636	004737	024354	15:	JSR	PC,LODFLT		;SETUP DEFAULT PARAMETERS
3281	005642	005037	001232		CLR	DRVSEL		;NO DRIVES SELECTED
3282	005646	005000			CLR	RO		;DETERMINE THE DRIVES THAT
3283	005650	012701	000001		MOV	#1,R1		;ARE AVAILABLE FOR TESTING
3284	005654	105760	034370	25:	TSTB	DRVSTA(RO)		
3285	005660	003403			BLE	35		
3286	005662	156037	034504	001232	BISB	ATABIT(RO),@DRVSEL		
3287	005670	005200		35:	INC	RO		
3288	005672	106301			ASLB	R1		
3289	005674	001367			BNE	25		
3290	005676	005037	034460	45:	CLR	@SEEKFG		;CLEAR SEEK FLAG
3291	005702	032737	000400	001220	BIT	#SW08,@C.SWR		;DO SEEK BEFORE DATA TRANSFER?
3292	005710	001002			BNE	55		;YES--BRANCH
3293	005712	005137	034460		COM	@SEEKFG		;NO
3294	005716	122737	000011	000041	55:	CMPB	#11,41	;LOADED FROM AN RPO4/5/6 ?
3295	005724	001003			BNE	105		;BR IF NOT

```

3296 005726 042737 000001 001232      BIC      #BIT00,DRVSEL ;CLEAR THE DRIVE 0 SELECTION BIT
3297 005734 104401 043233      105:    TYPE      ,DRIVES ;'DRIVES(S) TO BE TESTED'
3298 005740 005037 017550      CLR      @#SENDCT ;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      65 ;YES--BRANCH
3302 005754 104401 043264      TYPE      ,NONE ;'NONE'
3303
3304 005760 000137 017366      JMP      @#SEOP ;GO TO END OF PROGRAM
3305 005764 006201      65:    ASR      R1 ;REPORT THE DRIVES TO BE TESTED
3306 005766 103011      BCC      75
3307 005770 005237 017550      INC      @#SENDCT ;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      85 ;NO--BRANCH
3314 006006 104401 043271      TYPE      ,COMMA ;','
3315 006012 005200      75:    INC      R0 ;FORM DRIVE NUMBER
3316 006014 000763      BR       65
3317 006016 013737 017550 017542 85:    MOV      @#SENDCT,@#SEOPCT
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 043273      TYPE      ,NOCLOCK ;'NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED'
3323 006046 005737 001232      RSTRT1: TST      DRVSEL ;ANY DRIVES SELECTED ?
3324 006052 001002      BNE      15 ;BR IF YES
3325 006054 000137 004660      JMP      START2 ;GET DRIVE SELECTION ENTRY
3326 006060 005037 001254      15:    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 034370      TSTB     DRVSTA(R2) ;IS DESIRED DRIVE ON-LINE?
3338 006132 003005      BGT      15 ;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      15:    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 024770      JSR      PC,@#LDCMD ;LOAD COMMAND INTO DPB.B AND DPB.C
3347 006172 012737 017366 001252      MOV      #SEOP,@#BYPASS ;IF ERROR GO TO END OF PROGRAM
3348 006200 112737 000020 004105      MOV      #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      25 ;BR IF YES
3351 006216 105037 004105      CLRB     DPB.A+1 ;CLEAR THE 'FMT22' BIT
  
```



```

3352 006222 112737 000143 004106 25:  MOVB  #SETFORM,DPB.A+2 ;SET THE FORMAT BIT PER DPB.A+1
3353 006230 004037 025034      JSR   RO,@CALL.A ;GO EXECUTE THE COMMAND
3354 006234 112737 000107 004106  MOVB  #RECAL,@DPB.A+2 ;RECAL=COMMAND
3355 006242 004037 025034      JSR   RO,@CALL.A ;GO EXECUTE THE COMMAND
3356 006246 104401 043361      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 044042      TYPE  ,MSG.SP ;TYPE SPACES
3362 006264 104401 043403      TYPE  ,SERIAL ;'SERIAL NUMBER'
3363 006270 012700 000004      MOV   #4,RO ;FOUR DIGITS TO TYPE
3364 006274 013701 004234      MOV   RP,REG+30,R1 ;SERIAL NUMBER
3365 006300 005002      35:  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      45:  .WORD 0
3377 006332 104401 006330      TYPE  ,45 ;TYPE
3378 006336 005300      DEC   RO ;ALL DIGITS TYPED?
3379 006340 003357      BGT   35 ;NO -- BRANCH
3380 006342 104401 001215      TYPE  ,$CRLF
3381 006346 113737 001364 001115  MOVB  ERR.CT,$ERMAX ;SETUP MAX ERROR COUNT
  
```





```

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 TST0:
3446 006354 000240 NOP
3447 006354 033737 001424 001234 BIT @#BITS+(0*2), TSTNMS ; DO THIS TEST?
3448 006356 001002 BNE 645 ; YES--BRANCH
3449 006366 000137 006516 JMP TST1 ; NO--GO TO THE NEXT TEST
3450 006372 012737 000000 001102 645: MOV #0, @#TSTNM ; SET UP TEST NUMBER AND
3451 ; CLEAR THE ERROR FLAG (SERFLG)
3452 006400 004737 024612 JSR PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
3453 006404 012737 006500 001110 MOV @TEST0, @#SLPERR ; 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 MOV# @25, @SERMAX ; MAX ERRORS ALLOWED FOR TEST
3457 006434 112737 000107 004106 MOV# @RECAL, @#DPB. A+2 ; RECAL=COMMAND
3458 006442 113737 001524 004134 MOV# @#FS, @#DPB. B+10 ; FS
3459 006450 113737 001516 004135 MOV# @#FT, @#DPB. B+11 ; FT
3460 006456 013737 001512 004136 MOV @#LC, @#DPB. B+12 ; LC
3461 006464 012737 006514 001252 MOV @EXITO, @#BYPASS ; GO TO EXITO ON ERROR
3462 006472 012737 006500 001106 MOV @TEST0, @#PADR ; SETUP LOOP ADDRESS
3463 006500 012706 001100 TESTO: MOV @STACK, SP ; SET UP STACK POINTER
3464 006504 004037 025034 JSR RO, @#CALL. A ; GO EXECUTE THE COMMAND
3465 006510 004037 025146 JSR RO, @#CALL. B ; GO EXECUTE THE COMMAND
3466 006514 000004 EXITO: 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 TST1:
3480 006516 000240 NOP
3481 006520 033737 001426 001234 BIT @#BITS+(1*2), TSTNMS ; DO THIS TEST?
3482 006526 001002 BNE 645 ; YES--BRANCH
3483 006530 000137 006674 JMP TST2 ; NO--GO TO THE NEXT TEST
3484 006534 012737 000001 001102 645: MOV #1, @#TSTNM ; SET UP TEST NUMBER AND
3485 ; CLEAR THE ERROR FLAG (SERFLG)
3486 006542 004737 024612 JSR PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
3487 006546 012737 006656 001110 MOV @TEST1, @#SLPERR ; 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 MOV# @25, @SERMAX ; MAX ERRORS ALLOWED FOR TEST
3491 006576 113737 001524 004134 MOV# @#FS, @#DPB. B+10 ; FS
3492 006604 113737 001526 004154 MOV# @#LS, @#DPB. C+10 ; LS
3493 006612 113737 001516 004135 MOV# @#FT, @#DPB. B+11 ; FT
    
```



```

3494 006620 113737 001520 004155      MOV      @@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 025336      JSR      RO, @CALL. C ;GO EXECUTE THE COMMAND
3501 006666 004037 025146      JSR      RO, @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 TST2:
3518 006674 000240      NOP
3519 006676 033737 001430 001234      BIT      @@BITS+<2*2>, TSTNMS ;DO THIS TEST?
3520 006704 001002      BNE      645 ;YES--BRANCH
3521 006706 000137 007112      JMP      TST3 ;NO--GO TO THE NEXT TEST
3522 006712 012737 000002 001102 645:  MOV      @2, @STSTNM ;SET UP TEST NUMBER AND
3523 ; * CLEAR THE ERROR FLAG (SERFLG)
3524 006720 004737 024612      JSR      PC, LODPRM ;LOAD THE PARAMETERS FOR THE TEST
3525 006724 012737 007004 001110      MOV      @TEST2, @SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
3526 006732 013777 001102 172202      MOV      @STSTNM, @DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3527 006740 013737 001506 001204      MOV      @ARPT, @TIMES ;GET THE ITERATION COUNT
3528 006746 112737 000031 001115      MOV      @25, @SERMAX ;MAX ERRORS ALLOWED FOR TEST
3529 006754 012737 006762 001106      MOV      @15, @LPADR ;SETUP LOOP ADDRESS
3530 006762 113737 001524 004134 15:  MOV      @@FS, @DPB. B+10 ;FS
3531 006770 113737 001516 004135      MOV      @@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      @, @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3535 007020 012706 001100      MOV      @STACK, SP ;LOAD THE STACK POINTER
3536
3537 INCSK:
3538 007030 063737 001514 004136      JSR      RO, @CALL. B ;GO EXECUTE THE COMMAND
3539 007036 023737 001512 004136      ADD      @@IC, @DPB. B+12 ;MOVE TO NEXT CYLINDER
3540 007044 002367      CMP      @@LC, @DPB. B+12 ;OUT OF CYLINDERS?
3541 007046 013737 001512 004136      BGE      INCSK ;NO--BRANCH
3542 007054 012737 007054 001110      MOV      @, @SLPERR ;SETUP THE ERROR LOOP ADDRESS
3543 007062 012706 001100      MOV      @STACK, SP ;LOAD THE STACK POINTER
3544
3545 DECSK:
3546 007066 004037 025146      JSR      RO, @CALL. B ;GO EXECUTE THE COMMAND
3547 007072 163737 001514 004136      SUB      @@IC, @DPB. B+12
3548 007100 023737 001510 004136      CMP      @@FC, @DPB. B+12
3549 007110 000004      BLE      DECSK
EXIT2:  SCOPE ;LOOP
    
```



```

3550
3551 ;*****
3552 ;*TEST 3          STEPPING SEEK TEST
3553
3554 ;*      THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4,
3555 ;*      8,16,32,64,128, AND 256.  AT THE COMPLETION OF EACH SEEK
3556 ;*      COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER
3557 ;*      OPERATION.
3558
3559 ;*****
3560 TST3:
3561 007112 000240      NOP
3562 007114 033737 001432 001234      BIT      @#BITS+<3*2>,TSTNMS ;DO THIS TEST?
3563 007122 001002      BNE      645          ;YES--BRANCH
3564 007124 000137 007306      JMP      TST4          ;NO--GO TO THE NEXT TEST
3565 007130 012737 000003 001102 645:  MOV      #3,@#TSTNM      ;SET UP TEST NUMBER AND
3566 ;CLEAR THE ERROR FLAG (SERFLG)
3567 007136 004737 024612      JSR      PC,LODPRM      ;LOAD THE PARNETERS FOR THE TEST
3568 007142 012737 007222 001110      MOV      @TEST3,@$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
3569 007150 013777 001102 171764      MOV      $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3570 007156 013737 001506 001204      MOV      @RPT,$TIMES    ;GET THE ITERATION COUNT
3571 007164 112737 000031 001115      MOVB    #25,$SERMAX     ;MAX ERRORS ALLOWED FOR TEST
3572 007172 012737 007200 001106      MOV      #15,$LPADR     ;SETUP TEST LOOP ADDRESS
3573 007200 113737 001524 004134 15:  MOVB    @#FS,@DPB.B+10 ;FS
3574 007206 113737 001516 004135      MOVB    @#FT,@DPB.B+11 ;FT
3575 007214 012737 007304 001252      MOV      @EXIT3,@BYPASS ;GO TO BYPASS ON ERROR
3576 007222 013737 001510 004136 TEST3: MOV      @#FC,@DPB.B+12 ;FC
3577 007230 012737 007230 001110      MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
3578 007236 012706 001100      MOV      @STACK,SP     ;LOAD THE STACK POINTER
3579 007242 004037 025146      JSR      RO,@CALL.B    ;GO EXECUTE THE COMMAND
3580 007246 013701 001514      MOV      IC,R1         ;CYLINDER 1
3581 007252 012737 007252 001110      MOV      #,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
3582 007260 012706 001100      MOV      @STACK,SP     ;LOAD THE STACK POINTER
3583 007264 010137 004136 15:  MOV      R1,@DPB.B+12  ;DESIRED CYLINDER
3584 007270 004037 025146      JSR      RO,@CALL.B    ;GO EXECUTE THE COMMAND
3585 007274 006301      ASL      R1            ;MOVE TO NEXT CYLINDER
3586 007276 020137 001512      CMP      R1,@LC        ;DONE?
3587 007302 003770      BLE     15            ;NO--LOOP
3588 007304 00J004      EXIT3:  SCOPE
3589
3590 ;*****
3591 ;*TEST 4          OSCILLATING SEEK TEST
3592
3593 ;*      THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
3594 ;*      TO "FC".  "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
3595 ;*      "LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC".  AT THE
3596 ;*      COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
3597 ;*      EXAMINED TO ENSURE PROPER OPERATION.
3598
3599 ;*****
3600 TST4:
3601 007306 000240      NOP
3602 007310 033737 001434 001234      BIT      @#BITS+<4*2>,TSTNMS ;DO THIS TEST?
3603 007316 001002      BNE      645          ;YES--BRANCH
3604 007320 000137 007674      JMP      TST5          ;NO--GO TO THE NEXT TEST
3605 007324 012737 000004 001102 645:  MOV      #4,@#TSTNM      ;SET UP TEST NUMBER AND
    
```





```

3662 007660 163701 001514      65:  SUB    @#IC,R1      ;NEXT CYLINDER
3663 007664 020137 001510      CMP    R1,@#FC        ;DONE?
3664 007670 002340                BGE    45              ;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    645            ;YES--BRANCH
3684 007706 000137 010074                JMP    TST6           ;NO--GO TO THE NEXT TEST
3685 007712 012737 000005 001102      645:  MOV    #5,@#TSTNM     ;SET UP TEST NUMBER AND
3686                                ;CLEAR THE ERROR FLAG (SERFLG)
3687 007720 004737 024612                JSR    PC,LODPRM      ;LOAD THE PARAMETERS FOR THE TEST
3688 007724 012737 010004 001110      MOV    #TST5,@#SLPERR ;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,$TIMES   ;GET THE ITERATION COUNT
3691 007746 112737 000031 001115      MOVB  #25,$SERMAX     ;MAX ERRORS ALLOWED FOR TEST
3692 007754 012737 007762 001106      MOV    #15,$SLPADR    ;SETUP LOOP ADDRESS
3693 007762 113737 001524 004134      15:  MOVB  @#FS,@#DPB.B+10 ;FS
3694 007770 113737 001516 004135      MOVB  @#FT,@#DPB.B+11 ;FT
3695 007776 012737 010072 001252      MOV    #EXITS,@#BYPASS ;GO TO EXITS ON ERROR
3696 010004 013701 001510                TEST5: MOV    @#FC,R1        ;START NC1 AT FC
3697 010010 013702 001512                MOV    @#LC,R2        ;START NC2 AT LC
3698 010014 012737 010014 001110      MOV    #,$SLPERR      ;SETUP THE ERROR LOOP ADDRESS
3699 010022 012706 001100                MOV    #STACK,SP      ;LOAD THE STACK POINTER
3700 010026 010137 004136                15:  MOV    R1,@#DPB.B+12  ;NC1
3701 010032 004037 025146                JSR    R0,@#CALL.B    ;GO EXECUTE THE COMMAND
3702 010036 010237 004136                MOV    R2,@#DPB.B+12  ;NC2
3703 010042 004037 025146                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    EXITS          ;YES--BRANCH
3708 010064 020237 001510                CMP    R2,@#FC ;?
3709 010070 002356                BGE    15              ;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 UNTIL "LC" IS
    
```

```
3718 ;* EXCEEDED BY ANY OF THE ABOVE VALUES. THE INITIAL VALUE OF "NC"
3719 ;* IS "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE
3720 ;* PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
3721
3722 ; ; *****
3723 TST6:
3724 010074 000240 NOP
3725 010076 033737 001440 001234 BIT @#BITS+(6*2),TSTNMS ;DO THIS TEST?
3726 010104 001002 BNE 645 ;YES--BRANCH
3727 010106 000137 010340 JMP TST7 ;NO--GO TO THE NEXT TEST
3728 010112 012737 000006 001102 645: MOV #6,@#TSTNM ;SET UP TEST NUMBER AND
3729 ;CLEAR THE ERROR FLAG (SERFLG)
3730 010120 004737 024612 JSR PC,LODPRM ;LOAD THE PARMETERS FOR THE TEST
3731 010124 012737 010204 001110 MOV #TST6,@#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
3732 010132 013777 001102 171002 MOV $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3733 010140 013737 001506 001204 MOV @RPT,$TIMES ;GET THE ITERATION COUNT
3734 010146 112737 000031 001115 MOVB #25,$SERMAX ;MAX ERRORS ALLOWED FOR TEST
3735 010154 012737 010162 001106 MOV #15,$LPADR ;SETUP LOOP ADDRESS
3736 010162 113737 001524 004134 15: MOVB @#FS,@#DPB.B+10 ;FS
3737 010170 113737 001516 004135 MOVB @#FT,@#DPB.B+11 ;FT
3738 010176 012737 010336 001252 MOV #EXIT6,@#BYPASS ;GO TO EXIT6 ON ERROR
3739 010204 013701 001510 TEST6: MOV @#FC,R1 ;PICKUP "FC"
3740 010210 013702 001512 MOV @#LC,R2 ;FORM LAST CYLINDER THAT
3741 010214 162702 000005 SUB #5,R2 ;IS AVAILABLE FOR TESTING
3742 010220 012737 010220 001110 MOV #,$SLPERR ;SETUP THE ERROR LOOP ADDRESS
3743 010226 012706 001100 MOV #STACK,SP ;LOAD THE STACK POINTER
3744 010232 020102 15: CMP R1,R2 ;LAST CYLINDER
3745 010234 003040 BGT EXIT6 ;YES--BRANCH
3746 010236 010137 004136 MOV R1,@#DPB.B+12 ;NC
3747 010242 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3748 010246 062737 000004 004136 ADD #4,@#DPB.B+12 ;NC+4
3749 010254 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3750 010260 162737 000003 004136 SUB #3,@#DPB.B+12 ;NC+1
3751 010266 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3752 010272 062737 000002 004136 ADD #2,@#DPB.B+12 ;NC+3
3753 010300 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3754 010304 162737 000001 004136 SUB #1,@#DPB.B+12 ;NC+2
3755 010312 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3756 010316 062737 000003 004136 ADD #3,@#DPB.B+12 ;NC+5
3757 010324 004037 025146 JSR RO,@#CALL.B ;GO EXECUTE THE COMMAND
3758 010330 063701 001514 ADD @#IC,R1
3759 010334 000736 BR 15
3760 010336 000004 EXIT6: SCOPE ;LOOP
3761
3762 ; ; *****
3763 ;*TEST 7 RANDOM SEEK TEST
3764
3765 ;* THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC'
3766 ;* 'LC'. AFTER EACH SEEK, THE POSITION OF THE DRIVE IS VERIFIED BY
3767 ;* READING A SECTOR FROM THE CURRENTLY ADDRESSED CYLINDER AND TRACK.
3768 ;* THE TRACK ADDRESS IS INCREMENTED FOR EACH SEEK SO THAT VERIFICATION
3769 ;* OF POSITIONING OCCURS USING EACH HEAD. TRACK ADDRESSES ARE INCREMENTED
3770 ;* BETWEEN PARAMTERS 'FT' AND 'LT'.
3771
3772 ; ; *****
3773 010340 TST7:
```



3774	010340	000240				NOP	
3775	010342	033737	001442	001234		BIT	2#BITS+(7*2), TSTNMS ; DO THIS TEST?
3776	010350	001002				BNE	645 ; YES--BRANCH
3777	010352	000137	010720			JMP	TST10 ; NO--GO TO THE NEXT TEST
3778	010356	012737	000007	001102	645:	MOV	#7, 2#TSTNM ; SET UP TEST NUMBER AND
3779							; CLEAR THE ERROR FLAG (SERFLG)
3780	010364	004737	024612			JSR	PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
3781	010370	012737	010462	001110		MOV	#TEST7, 2#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
3782	010376	013777	001102	170536		MOV	#TSTNM, 2#DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3783	010404	013737	001506	001204		MOV	2#RPT, #TIMES ; GET THE ITERATION COUNT
3784	010412	112737	000031	001115		MOVB	#25, #SERMAX ; 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, 2#DPB, A+2 ; SEEK=COMMAND
3787	010434	112737	000173	004126		MOVB	#READHD, DPB, B+2 ; READ HEADER & DATA COMMAND
3788	010442	013704	034516			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, SLPADR ; 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	15 ; BR IF THEY ARE
3795	010504	004737	023512			JSR	PC, SRAND ; CYCLE THE RANDOM NUMBER GENERATOR
3796	010510	013746	023610			MOV	#HNUM, -(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	023614			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 QUOTIENT
3804	010542	013737	004136	004116		MOV	DPB, B+12, DPB, A+12 ; COPY NEW CYLINDER ADDRESS
3805	010550				15:		
3806	010550	012737	010550	001110		MOV	#, SLPERR ; SETUP THE ERROR LOOP ADDRESS
3807	010556	012706	001100			MOV	#STACK, SP ; LOAD THE STACK POINTER
3808	010562	004037	025034			JSR	RD, 2#CALL, A ; GO EXECUTE THE COMMAND
3809	010566	012737	010566	001110		MOV	#, SLPERR ; 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	25 ; BR IF YES
3818	010626	105216				INCB	(SP) ; INCREMENT THE SECTOR ADDRESS
3819	010630	105216			25:	INCB	(SP) ; INCREMENT THE SECTOR ADDRESS
3820	010632	112637	004174			MOVB	(SP)+, DTADPB+10 ; LOAD THE DPB
3821	010636	013746	001630			MOV	PRMLT+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	45 ; BR IF NOT
3825	010652	103403				BLO	35 ; BR IF ADDRESS IS 2 GREATER
3826	010654	105037	004174			CLRB	DTADPB+10 ; RESET TO SECTOR ADDRESS 0
3827	010660	000403				BR	45 ; CONTINUE
3828	010662	112737	000001	004174	35:	MOVB	#1, DTADPB+10 ; RESET ADDRESS TO SECTOR 1
3829	010670				45:		

```

3830 010670 004037 025146 JSR RO, @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 ?
    
```

\*\*\*\*\*  
 ;\*TEST 10 SERVO SETTLE DOWN TEST

```

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+IC'
3844 ;* ('NC1' STARTS AT VALUE 'FC'). AT THE COMPLETION OF 1000 (10) SEEKS,
3845 ;* 'NC1' IS INCREMENTED BY VALUE 'IC' 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 RPO4, THE
3853 ;* OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 US), THE DRIVE
3854 ;* WILL REPORT A 'WRU' ERROR. (RPO5/6'S MAY ALSO REPORT 'MHS' 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.
    
```

\*\*\*\*\*  
 TST10:

```

3873 010720 000240 NOP
3874 010720 033737 001444 001234 BIT @#BITS+(10*2), TSTNMS ;DO THIS TEST?
3875 010722 001002 BNE 645 ;YES--BRANCH
3876 010730 000137 012024 JMP TST11 ;NO--GO TO THE NEXT TEST
3877 010732 012737 000010 001102 645: MOV #10, @#STSTNM ;SET UP TEST NUMBER AND
3878 ;CLEAR THE ERROR FLAG (SERFLG)
3879 010744 004737 024612 JSR PC, LODPRM ;LOAD THE PARMETERS FOR THE TEST
3880 010750 012737 011126 001110 MOV #TEST10, @#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
3881 010756 013777 001102 170156 MOV $TSTNM, @DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3882 010764 013737 001506 001204 MOV @RPT, $TIMES ;GET THE ITERATION COUNT
3883 010772 112737 000031 001115 MOVB #25, $ERMAX ;MAX ERRORS ALLOWED FOR TEST
3884 011000 012737 011006 001106 MOV #15, $LPADR ;SETUP THE LOOP ADDRESS
    
```



```

3886 011006
3887 011006 112737 000105 004106 15:      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 047714 004172      MOV       #BUFFER, DTADPB+6 ;BUFFER ADDRESS
3896 011074 005000      CLR       R0 ;PATTERN POINTER (WC PATTERN)
3897 011076 004737 030362      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 034516      MOV      RPADR, R4 ;UNIBUS ADDRESS OF THE RH11
3901 011114 004737 042520      JSR      PC, CLRQUE ;CLEAR THE OPERATION QUEUES
3902 011120 012737 012022 001252      MOV      #EXIT10, BYPASS ;ERROR EXIT FROM TEST
3903 011126
3904 011126 012737 011126 001110 TEST10:  MOV      #. , SLPERR ;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      35 ;BR IF NO CLOCK
3909 011154 100405      BMI     15 ;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       25 ;CONTINUE
3913 011170 017746 170212 15:      MOV      @LKV, -(SP) ;SAVE THE 'L' CLOCK VECTOR
3914 011174 013746 001406      MOV      LKV, -(SP) ;SAVE THE VECTOR ADDRESS
3915 011200 012776 011756 000000 25:      MOV      #TST10B, @ (SP) ;CHANGE THE VECTOR
3916 011206 012777 027054 023304 35:      MOV      #DORT1, @RPVEC ;CHANGE THE RPO4/RPOS VECTOR
3917 011214 012737 000010 001344      MOV      #8. , SEKTR ;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 45:      TSTB    RPDS1(R4) ;HAS THE DRIVE FINISHED ?
3925 011266 100402      BMI     55 ;BR IF IT HAS
3926 011270 000001      WAIT    ;WAIT FOR THE OPERATION TO COMPLETE
3927 011272 000773      BR      45 ;CONTINUE
3928 011274 012737 000340 177776 55:      MOV      #PR7, @#PS ;CHANGE PRIORITY TO MAX
3929 011302 032764 040000 000012      BIT      #BIT14, RPDS1(R4) ;ERROR ?
3930 011310 001412      BEQ     65 ;BR IF NOT
3931 011312 012702 004104      MOV      #DPB. A, R2 ;DPB POINTER
3932 011316 004737 042036      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 037400 023154 65:      MOV      #ISR, @RPVEC ;SETUP THE RPO4/RPOS 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	#, SLPERR	; 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	PRMLT+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)+, RPOA(R4)	; LOAD THE TRACK/SECTOR ADDR
3967	011514	005037	004202			CLR	DTADPB+16	; RESET 'DONE' INDICATOR
3968	011520	012737	004164	034430		MOV	#DTADPB, TRNSWT	; LOAD 'TRANSFER' DPB ADDRESS
3969	011526	010137	034502			MOV	R1, DTUW	; ADDRESS OF DRIVE TRANSFERRING
3970	011532	112761	000001	034360		MOVB	#1, DRVACT(R1)	; SET DRIVE ACTIVE INDICATOR
3971	011540	006301				ASL	R1	; SHIFT DRIVE ADDRESS
3972	011542	012761	001750	034462		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	025546			JSR	RO, 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 INCREMENT 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	023512			JSR	PC, \$RAND	; CYCLE THE RANDOM NUMBER GENERATOR
3994	011706	013746	023610			MOV	\$HNUM, -(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 023614 JSR PC,SDIV ;DIVIDE
3999 011726 062637 004116 ADD (SP)+,DPB.A+12 ;ADD THE REMAINDER TO THE INITIAL CYLINDER
4000 011732 005726 TST (SP)+ ;DISCARD THE QUOTIENT
4001 011734 023737 004116 004176 CMP DPB.A+12,DTADPB+12 ;SAME CYLINDER SELECTED AS LAST TIME ?
4002 011742 001754 BEQ 75 ;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 SEKTR ;DECREMENT THE SEEK TIMER
4006 011762 001016 BNE 15 ;CONTINUE IF NOT DONE
4007 011764 012702 004104 MOV #DPB.A,R2 ;DPB ADDRESS
4008 011770 004737 042036 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 MOV R1,RPCS2(R4) ;RESELECT THE DRIVE
4012 012010 016676 000002 000000 MOV 2(SP),2(SP) ;RESTORE THE CLOCK VECTOR ADDRESS
4013 012016 000401 BR EXIT10 ;ABORT THE TEST
4014 012020 000002 15: 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 CYLINDER '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 TST11:
4031 012024 000240 NOP
4032 012026 033737 001446 001234 BIT #BITS+<11*2>,TSTNMS ;DO THIS TEST?
4033 012034 001002 BNE 645 ;YES--BRANCH
4034 012036 000137 012244 JMP TST12 ;NO--GO TO THE NEXT TEST
4035 012042 012737 000011 001102 645: MOV #11,#STSTNM ;SET UP TEST NUMBER AND
4036 ;CLEAR THE ERROR FLAG (SERFLG)
4037 012050 004737 024612 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
4038 012054 012737 012164 001110 MOV #TST11,#SLPERR ;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,$TIMES ;GET THE ITERATION COUNT
4041 012076 112737 000031 001115 MOVB #25,$SERMAX ;MAX ERRORS ALLOWED FOR TEST
4042 012104 012737 012112 001106 MOV #15,$SLPADR ;SETUP THE LOOP ADDRESS
4043 012112 113737 001524 004134 15: 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 15:
4052 012170 004037 025336 JSR RO,#CALL.C ;GO EXECUTE THE COMMAND
4053 012174 004037 025146 JSR RO,#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	15	; BR IF NOT
4057	012216	013737	001510	004156	MOV	FC,DPB.C+12	; RESET ENDING CYLINDER 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	15	; BR IF NOT
4061	012242	000004			EXIT11:	SCOPE	; LOOP ?
4062							







4119	012402	012737	013000	001206		MOV	#EXIT12,SESCAPE	::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	027056			JSR	PC,@STRMR	::INITIALIZE THE TIMERS
4128	012446	012777	012654	166720		MOV	#75,@PKV	::SETUP VECTOR IN CASE OF OVERFLOW
4129	012454	012777	027054	022036		MOV	#DORT1,@RVEC	::SETUP RPO4/5/6 VECTOR
4130	012462	005077	166714		15:	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	25	::NO--BRANCH
4137	012520	104412				SAVREG		::SAVE R0-R5
4138	012522	012702	004164			MOV	#DTADPB,R2	::DPB POINTER
4139	012526	004737	042036			JSR	PC,@SVRH11	::SAVE ALL THE RH11/RPO4 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		25:	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	35	::NO--BRANCH
4151	012610	104412				SAVREG		::SAVE R0-R5
4152	012612	012702	004164			MOV	#DTADPB,R2	::DPB POINTER
4153	012616	004737	042036			JSR	PC,@SVRH11	::SAVE ALL THE RH11/RPO4 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	027122		35:	JSR	PC,@COUNT	::UPDATE THE COUNT
4159	012646	005301				DEC	R1	::DONE?
4160	012650	003304				BGT	15	::NO--BRANCH
4161	012652	000424				BR	85	::YES--GO TO THE EXIT
4162	012654	042777	000101	166516	75:	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	042036			JSR	PC,@SVRH11	::SAVE ALL THE RH11/RPO4 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				85:			
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 024076              JSR      PC,@ST.CLK      ;INITIALIZE THE CLOCK
4177 012744 012777 037400 021546      MOV      @ISR,@RVEC      ;RESTORE RH11/RPO4/5/6 INT. VECTOR
4178 012752 032737 000100 001220      BIT      @SW06,@C.SWR    ;60 HZ?
4179 012760 001004              BNE      95              ;NO -- BRANCH
4180 012762 004037 027254              JSR      RO,@TYPTIM      ;GO TYPE THE TIMES
4181 012766 002774              T7A                          ;POINTER
4182 012770 000403              BR       EXIT12          ;GO TO EXIT
4183 012772              95:
4184 012772 004037 027254              JSR      RO,@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 TST13:
4201 013002 000240      NOP
4202 013004 033737 001452 001234      BIT      @@BITS+<13*2>,TSTNMS ;DO THIS TEST?
4203 013012 001002              BNE      645            ;YES--BRANCH
4204 013014 000137 013446              JMP      TST14          ;NO--GO TO THE NEXT TEST
4205 013020 012737 000013 001102 645:      MOV      #13,@STSTNM    ;SET UP TEST NUMBER AND
4206 ;CLEAR THE ERROR FLAG (SERFLG)
4207 013026 004737 024612              JSR      PC,LODPRM      ;LOAD THE PARAMETERS FOR THE TEST
4208 013032 012737 013002 001110      MOV      @TST13,@SLPERR ;SETUP THE ERROR LOOP ADDRESS
4209 013040 013777 001102 166074      MOV      @STSTNM,@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      MOV      @25,@SERMAX    ;MAX ERRORS ALLOWED FOR TEST
4212 013062 005737 001244              TST      @CLKSTA        ;KW11-P CLOCK?
4213 013066 003002              BGT      15            ;YES--START TEST
4214 013070 000137 013446              JMP      TST14          ;NO--GO TO NEXT TEST
4215 013074 012737 013074 001106 15:      MOV      #15,@SLPADR    ;SETUP THE LOOP ADDRESS
4216 013102 004037 026672              JSR      RO,@SRCHOO     ;DO A MASSBUS INIT. AND RECAL
4217 013106 000402              BR       25            ;NO ERROR RETURN
4218 013110 000137 013444              JMP      EXIT13         ;ERROR RETURN--SCOPE LOOP CALL
4219 013114 012703 003014 25:      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 027056              JSR      PC,@STRMR      ;INITIALIZE THE TIMERS
4226 013152 012777 013340 166214      MOV      @75,@PKV       ;SETUP INCASE OF OVERFLOW
4227 013160 012777 027054 021332      MOV      @DORT1,@RVEC   ;SET RPO4/5/6 VECTOR
4228 013166 005077 166210 15:      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 042777 000101 166156      BIC      #101, @PKCS     ; STOP THE CLOCK
4234 013222 032764 040000 000012      BIT      @BIT14, RPDS1(R4) ; ANY DISK ERRORS?
4235 013230 001415                    BEQ      25              ; NO--BRANCH
4236 013232 104412                    SAVREG                    ; SAVE R0-R5
4237 013234 012702 004164                    MOV      @DTADPB, R2     ; DPB POINTER
4238 013240 004737 042036                    JSR      PC, @SVRH11    ; SAVE ALL THE RH11/RPO4 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 027122 25:          JSR      PC, @COUNT    ; COUNT THIS SEEKS TIME
4244 013270 004737 026450                    JSR      PC, @TWOMS     ; STALL FOR 2 MILLISECONDS
4245 013274 005705                    TST      R5              ; UP OR DOWN?
4246 013276 001011                    BNE     45              ; DOWN--BRANCH
4247 013300 005237 004176 35:          INC      @DTADPB+12     ; MOVE TO NEXT CYLINDER
4248 013304 023737 004176 001512      CMP      @DTADPB+12, LC ; OUT OF CYLINDERS?
4249 013312 002725                    BLT     15              ; NO--GO DO THE NEXT SEEK
4250 013314 012705 177777                    MOV      #-1, R5        ; SET UP/DOWN SWITCH TO DOWN
4251 013320 000722                    BR      15              ; GO DO THE NEXT SEEK
4252 013322 005337 004176 45:          DEC      @DTADPB+12     ; MOVE TO NEXT CYLINDER
4253 013326 023727 004176 000000      CMP      @DTADPB+12, #0 ; OUT OF CYLINDERS?
4254 013334 003314                    BGT     15              ; NO--GO DO THE NEXT SEEK
4255 013336 000424                    BR      85              ; GO TO THE EXIT
4256 013340 042777 000101 166032 75:      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 042036                    JSR      PC, @SVRH11    ; SAVE ALL THE RH11/RPO4 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 85:
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 024076                    JSR      PC, @ST.CLK    ; INITIALIZE THE CLOCK
4271 013430 012777 037400 021062      MOV      #ISR, @RVEC     ; RESTORE RH11/RPO4/5/6 INT. VECTOR
4272 013436 014037 027254                    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 MEASUREMENT 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 MEASUREMENT.
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 RPO4/5 OR TO 255 (10)

```



D 7

```

4287 ;* FOR AN RPO6.
4288
4289 ;; *****
4290 TST14:
4291 NOP
4292 BIT @#BITS+(14*2), TSTNMS ; DO THIS TEST?
4293 BNE 645 ; YES--BRANCH
4294 JMP TST15 ; NO--GO TO THE NEXT TEST
4295 MOV #14, @#TSTNM ; SET UP TEST NUMBER AND
4296 ; CLEAR THE ERROR FLAG (SERFLG)
4297 JSR PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
4298 MOV @TST14, @#SLPERR ; SETUP THE ERROR LOOP ADDRESS
4299 MOV @TSTNM, @DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4300 MOV @RPT, @TIMES ; GET THE ITERATION COUNT
4301 MOVB #25, @SERMAX ; MAX ERRORS ALLOWED FOR TEST
4302 TST @CLKSTA ; KW11-P CLOCK?
4303 BGT 15 ; YES--START TEST
4304 JMP TST15 ; NO--GO TO NEXT TEST
4305 MOV #15, @SLPADR ; SET THE LOOP ADDRESS
4306 JSR RO, @#SRCHOO ; DO A MASSBUS INIT & RECAL
4307 BR 25 ; RETURN HERE IF NO ERROR
4308 JMP EXIT14 ; RETURN HERE ON ERROR
4309 MOV @T11, R3 ; PARAMETER POINTER
4310 MOV @EXIT14, @ESCAPE ; ESCAPE TO EXIT14 ON ERROR
4311 TEST14: MOV @STACK, SP ; SETUP STACK
4312 MOV #128, R1 ; REPEAT "0-136-0" 128 TIMES
4313 JSR PC, @#STRMTR ; INIT. THE COUNTERS
4314 MOV #75, @PKV ; SET UP VECTOR IN CASE OF OVERFLOW
4315 MOV @DORT1, @RPVEC ; SETUP RPO4/5/6 VECTOR
4316 CLR @PKB ; START COUNT AT ZERO
4317 MOV LC, @PCA(R4) ; 'MIDDLE' CYLINDER
4318 MOV @SEEK, @PCS1(R4) ; START A SEEK
4319 MOV #131, @PKCS ; START THE CLOCK
4320 WAIT ; WAIT ON INTERRUPT
4321 BIC #101, @PKCS ; STOP CLOCK
4322 BIT @BIT14, @RPS1(R4) ; ERR=1?
4323 BEQ 25 ; NO--BRANCH
4324 SAVREG ; SAVE RO-R5
4325 MOV @DTADPB, R2 ; DPB POINTER
4326 JSR PC, @#SVRH11 ; SAVE ALL THE RH11/RPO4 REGISTERS
4327 MOV @BIT05, @PCS2(R4) ; MASSBUS CLEAR
4328 MOV @DTADPB, @PCS2(R4) ; SELECT DRIVE
4329 RESREG ; RESTORE RO-R5
4330 ERROR 17
4331 CLR R5 ; SET UP/DOWN SWITCH TO UP
4332 JSR PC, @#COUNT ; UPDATE THE COUNT
4333 JSR PC, @#TWOMS ; STALL FOR 2 MILLISECONDS
4334 CLR @PKB ; START THE COUNT AT ZERO
4335 MOV FC, @PCA(R4) ; BEGINNING CYLINDER
4336 MOV @SEEK, @PCS1(R4) ; START A SEEK
4337 MOV #131, @PKCS ; START THE CLOCK
4338 WAIT ; WAIT ON INTERRUPT
4339 BIC #101, @PKCS ; STOP THE CLOCK
4340 BIT @BIT14, @RPS1(R4) ; ERR=1?
4341 BEQ 35 ; NO--BRANCH
4342 SAVREG ; SAVE RO-R5
    
```

```

4343 014004 012702 004164      MOV      #DTADPB,R2      ;DPB POINTER
4344 014010 004737 042036      JSR      PC,@SVRH11     ;SAVE ALL THE RH11/RPO4 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 35:      MOV      @-1,R5        ;SET UP/DOWN SWITCH TO DOWN
4350 014040 004737 027122      JSR      PC,@COUNT    ;UPDATE THE COUNT
4351 014044 004737 026450      JSR      PC,@TWOMS     ;STALL FOR 2 MILLISECONDS
4352 014050 005301      DEC      R1            ;DONE?
4353 014052 003263      BGT      15           ;NO--BRANCH
4354 014054 000424      BR       85           ;YES--EXIT
4355 014056 042777 000101 165314 75:      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 042036      JSR      PC,@SVRH11     ;SAVE ALL THE RH11/RPO4 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      85:
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 024076      JSR      PC,@ST.CLK    ;INITIALIZE THE CLOCK
4370 014146 012777 037400 020344  MOV      #ISR,@RVEC     ;RESTORE RH11/RPO4/5/6 INT. VECTOR
4371 014154 004037 027254      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 RPO4/5'S AND TO 814 (10)
4385      ;*      FOR RPO6'S.
4386
4387      ;*****
4388      TST15:
4389 014164 000240      NOP
4390 014166 033737 001456 001234  BIT      @#BITS+(15*2),TSTNMS ;DO THIS TEST?
4391 014174 001002      BNE      645         ;YES--BRANCH
4392 014176 000137 014702      JMP      TST16       ;NO--GO TO THE NEXT TEST
4393 014202 012737 000015 001102 645:      MOV      #15,@STSTNM  ;SET UP TEST NUMBER AND
4394      ;CLEAR THE ERROR FLAG (SERFLG)
4395 014210 004737 024612      JSR      PC,LODPRM    ;LOAD THE PARAMETERS FOR THE TEST
4396 014214 012737 014164 001110  MOV      #TST15,@SLPERR ;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, SERMAX	; MAX ERRORS ALLOWED FOR TEST
4400	014244	005737	001244			TST	@CLKSTA	; K11-P CLOCK
4401	014250	003002				BGT	15	; YES--START TEST
4402	014252	000137	014702			JMP	TST16	; NO--GO TO NEXT TEST
4403	014256	012737	014256	001106	15:	MOV	#15, SLPADR	; SETUP THE LOOP ADDRESS
4404	014264	004037	026672			JSR	RO, @SRCHOO	; DO A MASSBUS INIT & RECAL
4405	014270	000402				BR	25	; RETURN HERE IF NO ERROR
4406	014272	000137	014700			JMP	EXIT15	; RETURN HERE ON ERROR
4407	014276	012703	003034		25:	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 "O-'LC'-O" 128 TIMES
4411	014320	004737	027056			JSR	PC, @STRMR	; INIT. THE TIMERS
4412	014324	012777	014574	165042		MOV	#75, @PKV	; SETUP VECTOR IN CASE OF OVERFLOW
4413	014332	012777	027054	020160		MOV	#DORT1, @RPVEC	; SETUP RPO4/5/6 VECTOR
4414	014340	005077	165036		15:	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	25	; NO--BRANCH
4422	014406	104412				SAVREG		; SAVE R0-R5
4423	014410	012702	004164			MOV	#DTADPB, R2	; DPB POINTER
4424	014414	004737	042036			JSR	PC, @SVRH11	; SAVE ALL THE RH11/RPO4 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			25:	CLR	R5	; SET THE UP/DOWN SWITCH TO UP
4430	014442	004737	027122			JSR	PC, @COUNT	; UP THE COUNT
4431	014446	004737	026450			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	35	; NO--BRANCH
4440	014520	104412				SAVREG		; SAVE R0-R5
4441	014522	012702	004164			MOV	#DTADPB, R2	; DPB POINTER
4442	014526	004737	042036			JSR	PC, @SVRH11	; SAVE ALL THE RH11/RPO4 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		35:	MOV	#-1, R5	; SET THE UP/DOWN SWITCH TO DOWN
4448	014556	004737	027122			JSR	PC, @COUNT	; UPDATE THE COUNT
4449	014562	004737	026450			JSR	PC, @TWOMS	; STALL FOR 2 MILLISECONDS
4450	014566	005301				DEC	R1	; DONE?
4451	014570	003263				BGT	15	; NO--BRANCH
4452	014572	000424				BR	85	; YES--EXIT
4453	014574	042777	000101	164576	75:	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 POINTF
4459 014620 004737 042036    JSR      PC,@SVRH11     ;SAVE ALL THE RH11/RPO4 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
4465 014644 012764 000040 000010 85:    MOV      @BIT05,RPCS2(R4) ;MASSBUS INIT.
4466 014652 013764 004164 000010    MOV      @DTADPB,RPCS2(R4) ;SELECT DRIVE
4467 014660 004737 024076    JSR      PC,@ST.CLK     ;INITIALIZE THE CLOCK
4468 014664 012777 037400 017626    MOV      @ISR,@RPVEC    ;RESTORE RH11/RPO4/5/6 INT. VECTOR
4469 014672 004037 027254    JSR      R0,@TYPTIM     ;GO TYPE THE TIMES
4470 014676 003034      T12                      ;POINTER
4471 014700 000004      EXIT15: SCOPE          ;LOOP ?

```



4472  
 4473  
 4474

.SBTTL \*\*\* ADDRESSING TESTS \*\*\*

```
;/# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# :
;/# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# :
;/# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# : /# :
```

;\*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  
 4491  
 4492  
 4493  
 4494  
 4495  
 4496  
 4497  
 4498  
 4499  
 4500  
 4501  
 4502  
 4503  
 4504  
 4505  
 4506  
 4507  
 4508  
 4509  
 4510  
 4511  
 4512  
 4513  
 4514  
 4515  
 4516  
 4517  
 4518  
 4519  
 4520  
 4521  
 4522  
 4523  
 4524  
 4525  
 4526  
 4527

;; \*\*\*\*\*  
 ;\*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.

;; \*\*\*\*\*  
 TST16:

```

NOP
BIT @#BITS+(16*2), TSTNMS ; DO THIS TEST?
BNE 645 ; YES--BRANCH
JMP TST17 ; NO--GO TO THE NEXT TEST
645: MOV #16, @#TSTNM ; SET UP TEST NUMBER AND
; CLEAR THE ERROR FLAG (SERFLG)
JSR PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
MOV #TEST16, @#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
MOV $TSTNM, @DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
MOV @RPT, $TIMES ; GET THE ITERATION COUNT
MOVB ERR. CT, $ERMAX ; MAX ERRORS ALLOWED FOR TEST
MOV #EXIT16, @#BYPASS
MOV #TEST16, $LPADR ; SETUP THE LOOP ADDRESS
TEST16: MOV #STACK, SP ; SET THE STACK POINTER
JSR PC, @#FILBUF ; FILL THE BUFFER WITH SECTOR ADDRESSES
MOV @#FC, @#DTADPB+12 ; CYLINDER
MOVB @#FT, @#DTADPB+11 ; TRACK
CLRB @#DTADPB+10 ; SECTOR
MOV TRCKWC, @#DTADPB+4 ; WORD COUNT
MOV #BUFFER, @#DTADPB+6 ; BUFFER ADDRESS
MOV #, $SLPERR ; SETUP THE ERROR LOOP ADDRESS
MOV #STACK, SP ; LOAD THE STACK POINTER
MOV #WRITE, @#DTADPB+2 ; COMMAND=WRITE DATA
JSR RO, @#DRVCAL ; START A DATA TRANSFER
MOV #WRCKD, @#DTADPB+2 ; COMMAND=WRITE CHECK DATA
    
```

```

4528 015074 012737 015074 001110      MOV      0., $LPERR      ; SETUP THE ERROR LOOP ADDRESS
4529 015102 012706 001100      MOV      $STACK, SP    ; LOAD THE STACK POINTER
4530 015106 004037 025526      JSR      RO, @ADRVCAL   ; START A DATA TRANSFER
4531 015112 012737 015112 001110      MOV      0., $LPERR      ; SETUP THE ERROR LOOP ADDRESS
4532 015120 012706 001100      MOV      $STACK, SP    ; LOAD THE STACK POINTER
4533 015124 004037 027744      JSR      RO, @ACLBUF    ; CLEAR BUFFER
4534 015130 012737 000171 004166      MOV      @READ, @DTADPB+2 ; COMMAND = READ
4535 015136 004037 025526      JSR      RO, @ADRVCAL   ; START A DATA TRANSFER
4536 015142 004037 030012      JSR      RO, @CKSCTR    ; CHECK THE SECTOR DATA READ
4537 015146 012700 047714      MOV      @BUFFER, RO    ; BUFFER ADDRESS
4538 015152 005001      CLR      R1            ; FIRST SECTOR
4539 015154 012737 015154 001110      MOV      0., $LPERR      ; SETUP THE ERROR LOOP ADDRESS
4540 015162 012706 001100      MOV      $STACK, SP    ; LOAD THE STACK POINTER
4541 015166 012737 000161 004166 15:      MOV      @WRITE, @DTADPB+2 ; COMMAND=WRITE DATA
4542 015174 012737 177400 004170      MOV      @SCTRNC, @DTADPB+4 ; WORD COUNT
4543 015202 010037 004172      MOV      RO, @DTADPB+6  ; BUFFER ADDRESS
4544 015206 110137 004174      MOV      R1, @DTADPB+10 ; SECTOR
4545 015212 004037 025526      JSR      RO, @ADRVCAL   ; START A DATA TRANSFER
4546 015216 012737 015216 001110      MOV      0., $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 047714 004172      MOV      @BUFFER, @DTADPB+6 ; BUFFER ADDRESS
4551 015252 105037 004174      CLR      @DTADPB+10    ; SECTOR
4552 015256 004037 025526      JSR      RO, @ADRVCAL   ; START A DATA TRANSFER
4553 015262 062700 001000      ADD      #512, RO      ; MOVE TO NEXT SECTOR
4554 015266 005201      INC      R1
4555 015270 023701 001630      CMP      PRMLMT+22, R1 ; DONE?
4556 015274 103334      BHS     15            ; 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      TST17:
4574      015300 000240      NOP
4575      015302 033737 001462 001234      BIT      @#BITS+(17*2), TSTNMS ; DO THIS TEST?
4576      015310 001002      BNE     645          ; YES--BRANCH
4577      015312 000137 015720      JMP     TST20        ; NO--GO TO THE NEXT TEST
4578      015316 012737 000017 001102 645:      MOV      #17, @#STSTNM ; SET UP TEST NUMBER AND
4579      ; CLEAR THE ERROR FLAG (SERFL)
4580      015324 004737 024612      JSR      PC, LODPRM   ; LOAD THE PARAMETERS FOR THE TEST
4581      015330 012737 015374 001110      MOV      @TEST17, @#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
4582      015336 013777 001102 163576      MOV      $STSTNM, @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, SERMAX ; MAX ERRORS ALLOWED FOR TEST
4585	015360	012737	015716	001252		MOV	#EXIT17, @BYPASS
4586	015366	012737	015374	001106		MOV	#TEST17, @SLPADR ; SETUP THE LOOP ADDRESS
4587	015374	012706	001100		TEST17:	MOV	#STACK, SP ; SET THE STACK POINTER
4588	015400	004737	027706			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	@STRWC, @DTADPB+4 ; WORD COUNT
4593	015434	012737	047714	004172		MOV	@BUFFER, @DTADPB+6 ; BUFFER ADDRESS
4594	015442	005000				CLR	RO ; TRACK=0
4595	015444	012737	015444	001110		MOV	R, @SLPERR ; SETUP THE ERROR LOOP ADDRESS
4596	015452	012706	001100			MOV	#STACK, SP ; LOAD THE STACK POINTER
4597	015456	110037	004175		15:	MOVB	RO, @DTADPB+11 ; TRACK ADDRESS
4598	015462	004037	025526			JSR	RO, @DRVCAL ; START A DATA TRANSFER
4599	015466	062737	001000	004172		ADD	#256. #2, @DTADPB+6 ; UPDATE BUFFER ADDRESS
4600	015474	005200				INC	RO ; UPDATE TRACK NUMBER
4601	015476	022700	000023			CMP	#19, RO ; OUT OF TRACKS?
4602	015502	003365				BGT	15 ; NO--BRANCH
4603	015504	012737	047714	004172		MOV	@BUFFER, @DTADPB+6 ; BUFFER ADDRESS
4604	015512	005000				CLR	RO
4605	015514	012737	015514	001110		MOV	R, @SLPERR ; 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		25:	MOVB	RO, @DTADPB+11 ; TRACK ADDRESS
4609	015540	004037	025526			JSR	RO, @DRVCAL ; START A DATA TRANSFER
4610	015544	062737	001000	004172		ADD	#256. #2, @DTADPB+6 ; UPDATE BUFFER ADDRESS
4611	015552	005200				INC	RO ; UPDATE TRACK NUMBER
4612	015554	022700	000023			CMP	#19, RO ; OUT OF TRACKS?
4613	015560	003365				BGT	25 ; NO--BRANCH
4614	015562	005000				CLR	RO ; FIRST TRACK ADDRESS
4615	015564	110037	004175		35:	MOVB	RO, @DTADPB+11 ; TRACK
4616	015570	010001				MOV	RO, R1 ; FORM BUFFER ADDRESS
4617	015572	012737	047714	004172		MOV	@BUFFER, @DTADPB+6 ; BUFFER ADDRESS
4618	015600	005301			45:	DEC	R1
4619	015602	002411				BLT	55
4620	015604	062737	001000	004172		ADD	#256. #2, @DTADPB+6
4621	015612	000772				BR	45
4622	015614	012737	015614	001110		MOV	R, @SLPERR ; SETUP THE ERROR LOOP ADDRESS
4623	015622	012706	001100			MOV	#STACK, SP ; LOAD THE STACK POINTER
4624	015626	012737	000161	004166	55:	MOV	@WRITE, @DTADPB+2 ; COMMAND=WRITE DATA
4625	015634	004037	025526			JSR	RO, @DRVCAL ; START A DATA TRANSFER
4626	015640	062737	001000	004172	65:	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	R, @SLPERR ; 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	025526			JSR	RO, @DRVCAL ; START A DATA TRANSFER
4632	015676	122737	000022	004175		CMPB	#18, @DTADPB+11 ; OUT OF TRACKS?
4633	015704	003365				BGT	65 ; NO--BRANCH
4634	015706	005200				INC	RO ; NEXT TRACK TO WRITE
4635	015710	022700	000022			CMP	#18, RO ; OUT OF TRACKS?
4636	015714	003323				BGT	35 ; NO--BRANCH
4637	015716	000004			EXIT17:	SCOPE	

K 7

4638  
4639  
4640  
4641  
4642  
4643  
4644  
4645  
4646  
4647  
4648  
4649  
4650  
4651  
4652  
4653  
4654  
4655  
4656  
4657  
4658  
4659  
4660  
4661  
4662  
4663  
4664  
4665  
4666  
4667  
4668  
4669  
4670  
4671  
4672  
4673  
4674  
4675  
4676  
4677  
4678  
4679  
4680  
4681  
4682  
4683  
4684  
4685  
4686  
4687  
4688  
4689  
4690  
4691  
4692  
4693

.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 17777 (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



4694	:	133333	004000	173777	155555	177757	066667	177777	000000
4695	:	165555	010000	167777	172666	177767	153333	177777	000000
4696	:	133333	020000	157777	155555	177773	066667	177777	000000
4697	:	165555	040000	137777	172666	177775	153333	177777	000000
4698	:	133333	100000	077777	155555	177776	066667	177777	000000
4699	:								
4700	:								

```
;; *****
4701
4702 015720 TST20:
4703 015720 000240 NOP
4704 015722 033737 001424 001236 BIT BITS+(20*2-40),TSTNMS+2 ;DO THIS TEST ?
4705 015730 001002 BNE 645 ;YES--BRANCH
4706 015732 000137 016440 JMP TST21 ;NO--GO TO THE NEXT TEST
4707 015736 012737 000020 001102 645: MOV #20,#STSTNM ;SET UP TEST NUMBER AND
4708 ;CLEAR THE ERROR FLAG (SERFLG)
4709 015744 004737 024612 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST
4710 015750 012737 016122 001110 MOV #TEST20,#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
4711 015756 013777 001102 163156 MOV STSTNM,#DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
4712 015764 013737 001506 001204 MOV #RPT,#TIMES ;GET THE ITERATION COUNT
4713 015772 113737 001364 001115 MOV#B ERR. CT,#SERMAX ;MAX ERRORS ALLOWED FOR TEST
4714 016000 012737 016000 001106 MOV #,#SLPADR ;SETUP THE LOOP ADDRESS
4715 016006 005000 CLR R0 ;CLEAR SWITCH
4716 016010 005004 CLR R4 ;FORM WORD COUNT IN R4
4717 016012 013701 001526 MOV #LS,R1
4718 016016 163701 001524 SUB #FS,R1
4719 016022 002004 BGE 15 ;BRANCH IF FS < OR = LS
4720 016024 063701 001630 ADD PRMLMT+22,R1 ;ADD MAXIMUM SECTOR ADDRESS TO
4721 016030 005201 INC R1 ;MAKE THE DIFFERENCE POSITIVE
4722 016032 005100 COM R0 ;SET SWITCH
4723 016034 062704 000400 15: ADD #256.,R4
4724 016040 005301 DEC R1
4725 016042 002374 BGE 15
4726 016044 005404 NEG R4
4727 016046 010405 MOV R4,R5 ;COPY NORMAL WORD COUNT INTO SMALL WC
4728 016050 005700 TST R0 ;SWITCH SET?
4729 016052 001412 BEQ 35 ;NO--BRANCH
4730 016054 005005 CLR R5 ;FORM WORD COUNT FOR LS < FS
4731 016056 013701 001630 MOV PRMLMT+22,R1
4732 016062 163701 001524 SUB #FS,R1
4733 016066 062705 000400 25: ADD #256.,R5
4734 016072 005301 DEC R1
4735 016074 002374 BGE 25
4736 016076 005405 NEG R5
4737 016100 113737 001524 004174 35: MOV#B #FS,#DTADPB+10 ;SECTOR
4738 016106 012737 047714 004172 MOV #BUFFER,#DTADPB+6 ;DATA BUFFER
4739 016114 012737 016436 001252 MOV #EXIT20,#BYPASS
4740 016122 012706 001100 TEST20: MOV #STACK,SP ;LOAD THE STACK POINTER
4741 016126 005037 001334 CLR #WCEFLG ;CLEAR THE WRITE CHECK ERROR FLAG
4742 016132 013701 001510 MOV #FC,R1 ;PICKUP FIRST CYLINDER
4743 016136 000407 BR 25
4744 016140 005720 15: TST (R0)+ ;MOVE TO NEXT DATA PATTERN
4745 016142 022700 000040 CMP #16.*2.,R0 ;OUT OF PATTERNS?
4746 016146 003004 BGT 35 ;NO--BRANCH
4747 016150 004037 027632 JSR R0,#INCCYL ;MOVE TO NEXT CYLINDER
4748 016154 000530 BR EXIT20 ;OUT OF CYLINDERS
4749 016156 005000 25: CLR R0 ;START WITH PATTERN 0
```

```

4750 016160 036037 001424 001530 35: BIT BITS(RO), @PAT ; THIS PATTERN SELECTED?
4751 016166 001764 BEQ 15 ; NO--BRANCH
4752 016170 013702 MOV @FT, R2 ; FIRST TRACK
4753 016174 010137 MOV R1, @DTADPB+12 ; CYLINDER
4754 016200 110237 004175 45: MOV 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 55 ; NO--BRANCH
4758 016216 022702 000022 CMP #18, R2 ; LAST DISK TRACK?
4759 016222 003002 BGT 55 ; NO--BRANCH
4760 016224 010537 004170 MOV R5, @DTADPB+4 ; SHORT WORD COUNT
4761 016230 017703 162704 55: 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 75 ; YES--BRANCH
4765 016244 004737 030362 JSR PC, @SETBUF ; MOVE DATA PATTERN INTO THE BUFFER
4766 016250 032777 000020 162662 BIT @SW04, @SWR ; INHIBIT WRITE?
4767 016256 001012 BNE 65 ; 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 025526 JSR RO, @DRVCAL ; START A DATA TRANSFER
4772 016304 032777 000010 162626 65: BIT @SW03, @SWR ; INHIBIT WRITE CHECK?
4773 016312 001012 BNE 75 ; 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 025526 JSR RO, @DRVCAL ; START A DATA TRANSFER
4778 016340 005737 001334 75: TST @WCEFLG ; WRITE CHECK ERROR FLAG SET?
4779 016344 001404 BEQ 85 ; NO--BRANCH
4780 016346 032777 000001 162564 BIT @SW00, @SWR ; PERFORM READ AFTER FATAL "WCE"?
4781 016354 001424 BEQ 95 ; NO--BRANCH
4782 016356 032777 000004 162554 85: BIT @SW02, @SWR ; INHIBIT READ DATA AND SOFTWARE COMPARE?
4783 016364 001020 BNE 95 ; 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 025526 JSR RO, @DRVCAL ; START A DATA TRANSFER
4788 016412 032777 000002 162520 BIT @SW01, @SWR ; COMPARE THE DATA?
4789 016420 001002 BNE 95 ; NO--BRANCH
4790 016422 004737 030452 JSR PC, @DATCMP ; YES--DO IT
4791 016426 004037 027602 95: JSR RO, @INCTRK ; MOVE TO NEXT TRACK
4792 016432 000642 BR 15 ; OUT OF TRACKS GO TO NEXT PATTERN
4793 016434 000661 BR 45 ; LOOP
4794 016436 000004 EXIT20: SCOPE ; SCOPE LOOP
    
```



4795  
4796  
4797  
4798  
4799  
4800  
4801  
4802  
4803  
4804  
4805  
4806  
4807  
4808  
4809  
4810  
4811  
4812  
4813  
4814  
4815  
4816  
4817  
4818  
4819  
4820  
4821  
4822  
4823  
4824  
4825  
4826  
4827  
4828  
4829  
4830  
4831  
4832  
4833  
4834  
4835  
4836  
4837  
4838  
4839  
4840  
4841  
4842  
4843  
4844  
4845  
4846  
4847  
4848  
4849  
4850

.SBTTL \*\*\* EXERCISE TEST \*\*\*  
 ; \*\*\*\*\*  
 ; TEST 21 RANDOM ADDRESS AND RANDOM PATTERN TEST  
 ; \* STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK  
 ; \* IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS  
 ; \* OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR  
 ; \* FOR THAT SECTOR.  
 ; \* THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES  
 ; \* "R" DEFAULTS TO 20,000.  
 ; \* 1) GENERATE A RANDOM ADDRESS  
 ; \* 2) WRITE A RANDOM PATTERN AT THE ADDRESS  
 ; \* GENERATED IN 1.  
 ; \* 3) GENERATE A RANDOM ADDRESS  
 ; \* 4) READ THE SECTOR AT THE ADDRESS  
 ; \* GENERATED IN 3.  
 ; \* 5) DO A SOFTWARE CHECK OF THE DATA READ IN 4.  
 ; \* 6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2  
 ; \* 7) GENERATE A RANDOM ADDRESS  
 ; \* 8) READ THE SECTOR AT THE ADDRESS  
 ; \* GENERATED IN 7.  
 ; \* 9) DO A SOFTWARE CHECK OF THE DATA READ IN 8  
 ; \* 10) DO A WRITE CHECK OF THE DATA WRITTEN IN 2

;; \*\*\*\*\*  
 TST21:  
 NOP  
 BIT BITS+(21\*2-40),TSTNMS+2 ;DO THIS TEST ?  
 BNE 645 ;YES--BRANCH  
 JMP TST22 ;NO--GO TO THE NEXT TEST  
 645: MOV #21,%STSTNM ;SET UP TEST NUMBER AND  
 ;CLEAR THE ERROR FLAG (SERFLG)  
 JSR PC,LODPRM ;LOAD THE PARAMETERS FOR THE TEST  
 MOV #TST21,%SLPERR ;SETUP THE LOOP ON ERROR ADDRESS  
 MOV STSTNM,%DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER  
 MOV %RPT,%TIMES ;GET THE ITERATION COUNT  
 MOVB ERR.CT,%SERMAX ;MAX ERRORS ALLOWED FOR TEST  
 MOV #,%SLPADR ;SETUP THE LOOP ADDRESS  
 MOV #EXIT21,%BYPASS  
 MOV #176543,%SHINUM ;PRIME THE RANDOM NUMBER GENERATOR  
 MOV #123456,%SLONUM  
 MOV %FC,%DTADPB+12 ;CYLINDER  
 MOV TRCKMC,%DTADPB+4 ;WORD COUNT  
 MOV %BUFFER,%DTADPB+6 ;BUFFER ADDRESS  
 MOV %WRITE,%DTADPB+2 ;COMMAND  
 BIT %SW15,%C.SWR ;WRITE THE DISK PACK BEFORE TESTING?  
 BNE 35 ;NO--BRANCH  
 JSR RO,%FILRAN ;FILL DATA BUFFER WITH RANDOM DATA  
 15: CLR %DTADPB+10 ;SECTOR AND TRACK  
 MOV #,%SLPERR ;SETUP THE ERROR LOOP ADDRESS  
 MOV %STACK,%SP ;LOAD THE STACK POINTER  
 25:  
 JSR RO,%DRVCAL ;START A DATA TRANSFER  
 INCB %DTADPB+11 ;NEXT TRACK



4851	016642	122737	000023	004175		CMPB	019, @DTADPB+11 ; TIME FOR NEXT CYLINDER
4852	016650	003370				BGT	25 ; NO--BRANCH
4853	016652	005237	004176			INC	@DTADPB+12
4854	016656	023737	001512	004176		CMF	@ALC, @DTADPB+12 ; OUT OF CYLINDERS?
4855	016664	002353				BGE	15 ; NO--BRANCH
4856	016666	012737	177400	004170	35:	MOV	@SCTRMC, @DTADPB+4 ; WORD COUNT
4857	016674	012737	016710	001106		MOV	@TEST21, @SLPADR
4858	016702	012737	016710	001110		MOV	@TEST21, @SLPERR
4859	016710	012706	001100		TEST21:	MOV	@STACK, SP ; SET STACK POINTER
4860	016714	004037	031244			JSR	RO, @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	047714			MOV	@BUFFER, R1 ; BUFFER ADDRESS
4865	016746	010137	004172			MOV	R1, @DTADPB+6
4866	016752	004037	031210			JSR	RO, @RANPAT ; GENERATE RANDOM PATTERN
4867	016756	012737	016756	001110		MOV	@SLPERR ; SETUP THE ERROR LOOP ADDRESS
4868	016764	012706	001100			MOV	@STACK, SP ; LOAD THE STACK POINTER
4869	016770	004037	025526			JSR	RO, @DRVCL ; START A DATA TRANSFER
4870	016774	004037	031244			JSR	RO, @RANADR
4871	017000	012737	000171	004166		MOV	@READ, @DTADPB+2 ; COMMAND=READ DATA
4872	017006	012737	050714	004172		MOV	@BUFFER+512, @DTADPB+6 ; BUFFER ADDRESS
4873	017014	012737	017014	001110		MOV	@SLPERR ; SETUP THE ERROR LOOP ADDRESS
4874	017022	012706	001100			MOV	@STACK, SP ; LOAD THE STACK POINTER
4875	017026	004037	025526			JSR	RO, @DRVCL ; START A DATA TRANSFER
4876	017032	004037	031012			JSR	RO, @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	047714	004172		MOV	@BUFFER, @DTADPB+6 ; DATA BUFFER ADDRESS
4881	017066	012737	017066	001110		MOV	@SLPERR ; SETUP THE ERROR LOOP ADDRESS
4882	017074	012706	001100			MOV	@STACK, SP ; LOAD THE STACK POINTER
4883	017100	004037	025526			JSR	RO, @DRVCL ; START A DATA TRANSFER
4884	017104	004037	031244			JSR	RO, @RANADR ; GENERATE A RANDOM ADDRESS
4885	017110	012737	000171	004166		MOV	@READ, @DTADPB+2 ; COMMAND=READ
4886	017116	012737	050714	004172		MOV	@BUFFER+512, @DTADPB+6 ; DATA BUFFER ADDRESS
4887	017124	012737	017124	001110		MOV	@SLPERR ; SETUP THE ERROR LOOP ADDRESS
4888	017132	012706	001100			MOV	@STACK, SP ; LOAD THE STACK POINTER
4889	017136	004037	025526			JSR	RO, @DRVCL ; START A DATA TRANSFER
4890	017142	004037	031012			JSR	RO, @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	047714	004172		MOV	@BUFFER, @DTADPB+6 ; DATA BUFFER ADDRESS
4895	017176	012737	017176	001110		MOV	@SLPERR ; SETUP THE ERROR LOOP ADDRESS
4896	017204	012706	001100			MOV	@STACK, SP ; LOAD THE STACK POINTER
4897	017210	004037	025526			JSR	RO, @DRVCL ; START A DATA TRANSFER
4898	017214	000004			EXIT21:	SCOPE	; LOOP ?
4899							
4900						.SBTTL	*** RPO4 ACCESS TIME ADJUSTMENT TEST ***
4901							
4902						;;	*****
4903						;	*TEST 22 RPO4 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 RPO4 USING THE



```
4907 ; 8 DDU. THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
4908 ; 8 SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.
4909
4910 ;; *****
4911 TST22:
4912 017216 000240 NOP
4913 017220 033737 001430 001236 BIT BITS+(22*2-40), TSTNMS+2 ; DO THIS TEST ?
4914 017226 001002 BNE 645 ; YES--BRANCH
4915 017230 000137 017366 JMP SEOP ; NO--GO TO THE END OF THE PROGRAM
4916 017234 012737 000022 001102 645: MOV #22, @#STSTNM ; SET UP TEST NUMBER AND
4917 ; CLEAR THE ERROR FLAG (SERFLG)
4918 017242 004737 024612 JSR PC, LODPRM ; LOAD THE PARAMETERS FOR THE TEST
4919 017246 012737 017304 001110 MOV @TEST22, @#SLPERR ; SETUP THE LOOP ON ERROR ADDRESS
4920 017254 013777 001102 161660 MOV STSTNM, @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 MOV #100, $ERMAX ; MAX ERRORS ALLOWED FOR TEST
4923 017276 012737 017304 001106 MOV @TEST22, $LPADR ; SETUP THE LOOP ADDRESS
4924 017304 012706 001100 TEST22: MOV @STACK, SP ; SETUP THE STACK POINTER
4925 017310 013737 001512 004116 MOV LC, DPB. A+12 ; ENDING CYLINDER
4926 017316 112737 000105 004106 MOV #SEEK, @#DPB. A+2 ; SEEK=COMMAND
4927 017324 004037 025034 JSR RO, @#CALL. A ; GO EXECUTE THE COMMAND
4928 017330 004037 026366 JSR RO, STALL ; STALL
4929 017334 001360 . WORD STALL3 ; ADDRESS OF STALL VALUE
4930 017336 013737 001510 004116 MOV FC, DPB. A+12 ; STARTING CYLINDER
4931 017344 112737 000105 004106 MOV #SEEK, @#DPB. A+2 ; SEEK=COMMAND
4932 017352 004037 025034 JSR RO, @#CALL. A ; GO EXECUTE THE COMMAND
4933 017356 004037 026366 JSR RO, STALL ; STALL
4934 017362 001360 . WORD STALL3 ; ADDRESS OF STALL VALUE
4935 017364 000004 EXIT22: SCOPE ; LOOP ?
4936
4937
4938
4939 . SBTTL END OF PASS ROUTINE
4940
4941 ;; *****
4942 ; INCREMENT THE PASS NUMBER ($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 SEOP:
4948 017366 104401 017374 TYPE , 655 ; TYPE ASCIZ STRING
4949 017372 000410 BR 645 ; GET OVER THE ASCIZ
4950 ;; 655: . ASCIZ <CR><LF><LF>/END OF PASS/
4951 645:
4952 017414 005737 001232 TST @#DRVSEL ; ANY DRIVES SELECTED?
4953 017420 001434 BEQ 15 ; NO--BRANCH
4954 017422 104401 017430 TYPE , 675 ; TYPE ASCIZ STRING
4955 017426 000405 BR 665 ; GET OVER THE ASCIZ
4956 ;; 675: . ASCIZ / ON DRIVE/
4957 665:
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 , 695 ; TYPE ASCIZ STRING
```

```

4963 017456 000412          BR      685          ;; GET OVER THE ASCIZ
4964          ;; 695: .ASCIZ / ERRORS DETECTED=/
4965 017504          685:
4966 017504 013746 001112    MOV     @SERTTL, -(SP) ;; SAVE @SERTTL FOR TYPEOUT
4967 017510 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
4968 017512 005037 001112    15:    CLR     @SERTTL      ;; 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 003030          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   , 655        ;; TYPE ASCIZ STRING
4980 017560 000410          BR      645        ;; GET OVER THE ASCIZ
4981          ;; 655: .ASCIZ <CR><LF>/END OF TEST /
4982 017602          645:
4983 017602 104401 017632    TYPE   , $ENULL     ;; TYPE NULL CHARACTER
4984 017606 013700 000042    $GET42: MOV     @#42, R0 ;; GET MONITOR ADDRESS
4985 017612 001405          BEQ     $DOAGN      ;; BRANCH IF NO MONITOR
4986 017614 000005          RESET          ;; CLEAR THE WORLD
4987 017616 004710          $ENDAD: JSR     PC, (R0) ;; GO TO MONITOR
4988 017620 000240          NOP          ;; SAVE ROOM
4989 017622 000240          NOP          ;; FOR
4990 017624 000240          NOP          ;; ACT11
4991 017626          $DOAGN:
4992 017626 000137          JMP     @PC+       ;; RETURN
4993 017630 006102          $RTNAD: .WORD     RESTART
4994 017632 377 377 000 $ENULL: .BYTE    -1, -1, 0 ;; NULL CHARACTER STRING
4995          017636          .EVEN
    
```



4996  
4997  
4998  
4999  
5000  
5001  
5002  
5003  
5004  
5005  
5006  
5007  
5008  
5009  
5010  
5011  
5012  
5013  
5014  
5015  
5016  
5017  
5018  
5019  
5020  
5021  
5022  
5023  
5024  
5025  
5026  
5027  
5028  
5029  
5030  
5031  
5032  
5033  
5034  
5035  
5036  
5037  
5038  
5039  
5040  
5041  
5042  
5043  
5044  
5045  
5046  
5047  
5048  
5049  
5050  
5051

017636  
017636 104407  
017640 032777 000400 161272  
017646 001411  
017650 005737 001230  
017654 001406  
017656 013737 001420 001150  
017664 013737 001422 001152  
017672 105237 001103  
017676 001775  
017700 013777 001102 161234  
017706 032777 002000 161224  
017714 001402  
017716 104401 001210  
017722 005237 001112  
017726 011637 001116  
017732 162737 000002 001116  
017740 117737 161152 001114  
017746 032777 020000 161164  
017754 001004  
017756 004737 020056  
017762 104401 001215  
017766  
017766 005777 161146  
017772 100002  
017774 000000  
017776 104407  
020000 032777 001000 161132  
020006 001402  
020010 013716 001110  
020014 005737 001206  
020020 001402  
020022 013716 001206  
020026  
020026 023737 000042 000046  
020034 001001  
020036 000240  
020040 013737 001414 001150  
020046 013737 001416 001152

.SBTTL \*\*\* SYSMAC SUBROUTINES \*\*\*  
 .SBTTL ERROR HANDLER ROUTINE

```

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

```

$ERROR:
CKSWR          ;TEST FOR CHANGE IN SOFT-SWR
BIT            #SW08, @SWR      ;SEND ERROR MESSAGE TO TTY?
BEQ           7$              ;YES--BRANCH
TST           @#LPTAVL        ;IS THERE A LINE PRINTER AVAILABLE?
BEQ           7$              ;NO--BRANCH
MOV           @#LPS, @#STPS    ;YES--SETUP STATUS
MOV           @#LPB, @#STPB    ;AND BUFFER REG. 'S FOR LINE PRINTER
7$: INCB      $ERFLG          ;SET THE ERROR FLAG
BEQ           7$              ;DON'T LET THE FLAG GO TO ZERO
MOV           $TSTNM, @DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG
BIT           #BIT10, @SWR     ;BELL ON ERROR?
BEQ           1$              ;NO - SKIP
TYPE         , $BELL          ;RING BELL
1$: INC       $ERTTL          ;COUNT THE NUMBER OF ERRORS
MOV          (SP), $ERRPC      ;GET ADDRESS OF ERROR INSTRUCTION
SUB          #2, $ERRPC
MOVVB       @$ERRPC, $ITEMB    ;STRIP AND SAVE THE ERROR ITEM CODE
BIT         #BIT13, @SWR      ;SKIP TYPEOUT IF SET
BNE         20$              ;SKIP TYPEOUTS
JSR         PC, TYPERR        ;GO TO USER ERROR ROUTINE
TYPE       , $CRLF

20$:
2$: TST      @SWR              ;HALT ON ERROR
BPL        3$                ;SKIP IF CONTINUE
HALT       ;HALT ON ERROR!
CKSWR          ;TEST FOR CHANGE IN SOFT-SWR
BIT        #BIT09, @SWR      ;LOOP ON ERROR SWITCH SET?
BEQ        4$                ;BR IF NO
MOV        $LPERR, (SP)      ;FUJGE RETURN FOR LOOPING
4$: TST     $ESCAPE          ;CHECK FOR AN ESCAPE ADDRESS
BEQ        5$                ;BR IF NONE
MOV        $ESCAPE, (SP)    ;FUJGE RETURN ADDRESS FOR ESCAPE
5$:
CMP        @#42, @#46        ;ACT11 AUTOMATIC MODE?
BNE        6$                ;NO, CONTINUE
NOP
6$: MOV     @#TPS, @#STPS    ;SET STATUS AND BUFFER REG. 'S
MOV        @#TPB, @#STPB    ;FOR TTY
    
```

```

5052 020054 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 APPROPRIATE INFORMATION
5059                      ;CONCERNING THE ERROR.
5060                      ;CALL
5061                      ;      JSR      PC, @#TYPERR
5062                      ;      RETURN
5063
5064 020056 113737 001102 001176 TYPERR: MOVB  @#STSTNL @#STMPO ;SAVE THE TEST NUMBER
5065 020064 104412          SAVREG          ;SAVE R0 - R5
5066 020066 162700 000004          SUB      #4, R0          ;FORM TEST PC
5067 020072 010037 001162          MOV      R0, @#SREG0      ;COPY R0-R5 IN $REG0-$REG5
5068 020076 010137 001164          MOV      R1, @#SREG1
5069 020102 010237 001166          MOV      R2, @#SREG2
5070 020106 010337 001170          MOV      R3, @#SREG3
5071 020112 010437 001172          MOV      R4, @#SREG4
5072 020116 010537 001174          MOV      R5, @#SREG5
5073 020122 113700 001114          MOVB   @#ITEMB, R0      ;PICKUP ERROR ITEM NUMBER
5074 020126 010001          MOV      R0, R1          ;AND COPY IT INTO R1
5075 020130 005300          DEC      R0              ;FORM INDEX FOR ERROR TABLE
5076 020132 106300          ASLB   R0
5077 020134 106300          ASLB   R0
5078 020136 106300          ASLB   R0
5079 020140 103002          BCC     15              ;IS ERROR > 37?
5080 020142 062700 000240          ADD     @#ITEM41-$ERRTB, R0 ;YES--FORM OFFSET
5081 020146 062700 004306          15:    ADD     @#ERRTB, R0      ;FORM ADDRESS
5082 020152 012037 020166          MOV     (R0)+, 25        ;GET ERROR MESSAGE (EM) POINTER
5083 020156 001447          BEQ     75              ;BRANCH IF THERE ISN'T ONE
5084 020160 104401 001215          TYPE   , $CRLF          ;"CARRIAGE RETURN - LINE FEED
5085 020164 104401          TYPE
5086 020166 000000          25:    .WORD  0              ;"EM" POINTER GOES HERE
5087 020170 162701 000041          SUB     #41, R1          ;SPECIAL ERROR ITEM NUMBER?
5088 020174 100440          BMI     75              ;NO--BRANCH
5089 020176 013701 001260          MOV     @#SVSTAT, R1     ;GET STATUS/ERROR INDICATOR
5090 020202 106301          ASLB   R1              ;STRIP "DONE" BIT (BIT07)
5091 020204 006301          ASL    R1              ;STRIP "ERROR" BIT (BIT15)
5092 020206 012702 004254          MOV     @#STATBL, R2     ;1ST ADDRESS ON STATUS MESSAGE POINTERS
5093 020212 005003          CLR    R3              ;CARRIAGE RETURN-LINE FEED SWITCH
5094 020214 104401 020222          TYPE   , 65$          ;; TYPE ASCIZ STRING
5095 020220 000402          BR     64$              ;; GET OVER THE ASCIZ
5096                      ;; 65$: .ASCIZ / (/
5097 020226          64$:
5098 020226 012237 020250          35:    MOV     (R2)+, 55        ;MESSAGE POINTER
5099 020232 006301          ASL    R1              ;TYPE THIS MESSAGE?
5100 020234 103013          BCC    65              ;NO--BRANCH
5101 020236 005103          COM    R3              ;YES--TYPE A "CR" & "LF"?
5102 020240 001002          BNE    45              ;NO--BRANCH
5103 020242 104401 001215          TYPE   , $CRLF          ;YES
5104 020246 104401          45:    TYPE
5105 020250 000000          55:    .WORD  0              ;MESSAGE POINTER GOES HERE
5106 020252 005701          TST    R1              ;MORE TO TYPE?
5107 020254 001403          BEQ    65              ;NO--BRANCH
    
```



```

5108 020256 104401 044042          TYPE      ,MSG. SP      ;YES--SPACES
5109 020262 000761                   BR          35          ;LOOP
5110 020264 001360                   BNE         35          ;BRANCH IF NOT FINISHED
5111 020266 104401 020274          TYPE      ,675         ;TYPE ASCIZ STRING
5112 020272 000401                   BR          665         ;GET OVER THE ASCIZ
5113                                     ;:675: .ASCIZ  /)/
5114 020276                                     665:
5115 020276 012037 020312          75:  MOV      (R0)+,R5      ;PICK UP DATA HEADER (DH) POINTER
5116 020302 001404                   BEQ         95          ;BRANCH IF NONE
5117 020304 104401 001215          TYPE      ,5CRLF       ;CARRIAGE RETURN-LINE FEED
5118 020310 104401                   TYPE
5119 020312 000000                   85:  .WORD    0          ;"DH" POINTER GOES HERE
5120 020314 012001                   95:  MOV      (R0)+,R1      ;PICKUP DATA TABLE (DT) POINTER
5121 020316 001450                   BEQ        205         ;BRANCH IF NONE
5122 020320 005005                   CLR         55          ;SET INDENT SWITCH
5123 020322 012000                   MOV      (R0)+,R0      ;DATA FORMAT (DF) POINTER
5124 020324 012002                   MOV      (R0)+,R2      ;NUMBER OF DH'S TO TYPE
5125 020326 001441                   BEQ        175         ;BRANCH IF DH NUMBER IS 0
5126 020330 005105                   COM         R5          ;NO INDENT
5127 020332 104401 001215          TYPE      ,5CRLF       ;CARRIAGE RETURN-LINE FEED
5128 020336 112003                   105: MOVB    (R0)+,R3      ;NUMBER OF DATA WORDS TO TYPE
5129 020340 112004                   MOVB    (R0)+,R4      ;AND HOW TO TYPE THEM
5130 020342 006004                   115: ROR     R4          ;OCTAL OR DECIMAL?
5131 020344 103403                   BCS      125         ;DECIMAL--BRANCH
5132 020346 013146                   MOV      @ (R1)+,-(SP) ;SAVE @ (R1)+ FOR TYPEOUT
5133 020350 104402                   TYPOC
5134 020352 000402                   BR        135         ;GO TYPE--OCTAL ASCII(ALL DIGITS)
5135 020354                                     125:
5136 020354 013146                   MOV      @ (R1)+,-(SP) ;SAVE @ (R1)+ FOR TYPEOUT
5137 020356 104405                   TYPDS
5138 020360 005303                   135: DEC     R3          ;GO TYPE--DECIMAL ASCII WITH SIGN
5139 020362 001403                   BEQ      145         ;MORE NUMBERS TO TYPE?
5140 020364 104401 044042          TYPE      ,MSG. SP      ;YES--TYPE SEPERATORS
5141 020370 000764                   BR        115         ;LOOP
5142 020372 005302                   145: DEC     R2          ;MORE DH'S?
5143 020374 003421                   BLE      205         ;NO--BRANCH
5144 020376 104401 001215          TYPE      ,5CRLF       ;YES--START A NEW LINE
5145 020402 005105                   COM         R5          ;INDENT?
5146 020404 001002                   BNE      155         ;NO--BRANCH
5147 020406 104401 044042          TYPE      ,MSG. SP      ;YES--TYPE SPACES
5148 020412 012037 020420          155: MOV      (R0)+,165    ;GET NEXT DH
5149 020416 104401                   TYPE
5150 020420 000000                   165: .WORD    0          ;DH POINTER GOES HERE
5151 020422 104401 001215          TYPE      ,5CRLF       ;CARRIAGE RETURN-LINE FEED
5152 020426 005705                   TST      R5          ;INDENT?
5153 020430 001342                   BNE      105         ;NO--BRANCH
5154 020432 104401 044042          175: TYPE      ,MSG. SP      ;YES--TYPE SPACES
5155 020436 000737                   BR        105         ;LOOP
5156 020440 104413                   205: RESREG  PC          ;RESTORE R0 - R5
5157 020442 000207                   RTS       PC          ;RETURN

```

5158  
 5159 .SBTTL TYPE ROUTINE

```

5160 ;:*****
5161 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
5162 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
5163
```

```

5164 ;NOTE1: SNULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
5165 ;NOTE2: SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
5166 ;NOTE3: SFILLC 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 ;#OR
5172 ;# TYPE
5173 ;# MESADR
5174 ;#
5175
5176 020444 105737 001157 STYPE: TSTB STPFLG ;; IS THERE A TERMINAL?
5177 020450 100002 BPL 15 ;; BR IF YES
5178 020452 000000 HALT ;; HALT HERE IF NO TERMINAL
5179 020454 000407 BR 35 ;; LEAVE
5180 020456 010046 15: MOV RO,-(SP) ;; SAVE RO
5181 020460 017600 000002 MOV @2(SP),RO ;; GET ADDRESS OF ASCIZ STRING
5182 020464 112046 25: MOVB (RO)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
5183 020466 001005 BNE 45 ;; BR IF IT ISN'T THE TERMINATOR
5184 020470 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
5185 020472 012600 605: MOV (SP)+,RO ;; RESTORE RO
5186 020474 062716 000002 35: ADD #2,(SP) ;; ADJUST RETURN PC
5187 020500 000002 RTI ;; RETURN
5188 020502 122716 000011 45: CMPB #HT,(SP) ;; BRANCH IF <HT>
5189 020506 001430 BEQ 85
5190 020510 122716 000200 CMPB #CRLF,(SP) ;; BRANCH IF NOT <CRLF>
5191 020514 001006 BNE 55
5192 020516 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
5193 020520 104401 TYPE ;; TYPE A CR AND LF
5194 020522 001215 SCRLF
5195 020524 105037 020660 CLRB SCHARCNT ;; CLEAR CHARACTER COUNT
5196 020530 000755 BR 25 ;; GET NEXT CHARACTER
5197 020532 004737 020614 55: JSR PC,STYPEC ;; GO TYPE THIS CHARACTER
5198 020536 123726 001156 65: CMPB SFILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
5199 020542 001350 BNE 25 ;; IF NO GO GET NEXT CHAR.
5200 020544 013746 001154 MOV SNULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
5201 ;; AND THE NULL CHAR.
5202 020550 105366 000001 75: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
5203 020554 002770 BLT 65 ;; BR IF NO--GO POP THE NULL OFF OF STACK
5204 020556 004737 020614 JSR PC,STYPEC ;; GO TYPE A NULL
5205 020562 105337 020660 DECB SCHARCNT ;; DO NOT COUNT AS A COUNT
5206 020566 000770 BR 75 ;; LOOP
5207
5208 ;HORIZONTAL TAB PROCESSOR
5209
5210 020570 112716 000040 85: MOVB #' ,(SP) ;; REPLACE TAB WITH SPACE
5211 020574 004737 020614 95: JSR PC,STYPEC ;; TYPE A SPACE
5212 020600 132737 000007 020660 BITB #7,SCHARCNT ;; BRANCH IF NOT AT
5213 020606 001372 BNE 95 ;; TAB STOP
5214 020610 005726 TST (SP)+ ;; POP SPACE OFF STACK
5215 020612 000724 BR 25 ;; GET NEXT CHARACTER
5216 020614 105777 160330 STYPEC: TSTB @STPS ;; WAIT UNTIL PRINTER IS READY
5217 020620 100375 BPL STYPEC
5218 020622 116677 000002 160322 MOVB 2(SP),@STPB ;; LOAD CHAR TO BE TYPED INTO DATA REG.
5219 020630 122766 000015 000002 CMPB #CR,2(SP) ;; IS CHARACTER A CARRIAGE RETURN?
  
```



```

5220 020636 001003          BNE      15          ;; BRANCH IF NO
5221 020640 105037 020660  CLRB    SCHARCNT    ;; YES--CLEAR CHARACTER COUNT
5222 020644 000406          BR      STYPEX      ;; EXIT
5223 020646 122766 000012 000002 15:  CMPB    #LF,2(SP)    ;; IS CHARACTER A LINE FEED?
5224 020654 001402          BEQ     STYPEX      ;; BRANCH IF YES
5225 020656 105227          INCB   (PC)+        ;; COUNT THE CHARACTER
5226 020660 000000  SCHARCNT: .WORD 0    ;; CHARACTER COUNT STORAGE
5227 020662 000207  STYPEX: 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          ;; $STYPON---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          ;; $STYPOC---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 020664 017646 000000  STYPOS: MOV     @2(SP),-(SP)    ;; PICKUP THE MODE
5256 020670 116637 000001 021107  MOVB    1(SP),SOFILL    ;; LOAD ZERO FILL SWITCH
5257 020676 112637 021111  MOVB    (SP)+,SOMODE+1  ;; NUMBER OF DIGITS TO TYPE
5258 020702 062716 000002  ADD     #2,(SP)        ;; ADJUST RETURN ADDRESS
5259 020706 000406  BR      STYPON
5260 020710 112737 000001 021107  STYPOC: MOVB    #1,SOFILL    ;; SET THE ZERO FILL SWITCH
5261 020716 112737 000006 021111  MOVB    #6,SOMODE+1    ;; SET FOR SIX(6) DIGITS
5262 020724 112737 000005 021106  STYPON: MOVB    #5,SOCNT    ;; SET THE ITERATION COUNT
5263 020732 010346  MOV     R3,-(SP)      ;; SAVE R3
5264 020734 010446  MOV     R4,-(SP)      ;; SAVE R4
5265 020736 010546  MOV     R5,-(SP)      ;; SAVE R5
5266 020740 113704 021111  MOVB    SOMODE+1,R4    ;; GET THE NUMBER OF DIGITS TO TYPE
5267 020744 005404  NEG     R4
5268 020746 062704 000006  ADD     #6,R4          ;; SUBTRACT IT FOR MAX. ALLOWED
5269 020752 110437 021110  MOVB    R4,SOMODE      ;; SAVE IT FOR USE
5270 020756 113704 021107  MOVB    SOFILL,R4      ;; GET THE ZERO FILL SWITCH
5271 020762 016605 000012  MOV     12(SP),R5     ;; PICKUP THE INPUT NUMBER
5272 020766 005003  CLR     R3            ;; CLEAR THE OUTPUT WORD
5273 020770 006105 15:  ROL     R5          ;; ROTATE MSB INTO "C"
5274 020772 000404  BR      35            ;; GO DO MSB
5275 020774 006105 25:  ROL     R5          ;; FORM THIS DIGIT
    
```

```

5276 020776 006105          ROL    R5
5277 021000 006105          ROL    R5
5278 021002 010503          MOV    R5,R3
5279 021004 006103          35:   ROL    R3          ;; GET LSB OF THIS DIGIT
5280 021006 105337 021110   DECB   $OMODE        ;; TYPE THIS DIGIT?
5281 021012 100016          BPL    75            ;; BR IF NO
5282 021014 042703 177770   BIC    #177770,R3    ;; GET RID OF JUNK
5283 021020 001002          BNE    45            ;; TEST FOR 0
5284 021022 005704          TST    R4            ;; SUPPRESS THIS 0?
5285 021024 001403          BEQ    55            ;; BR IF YES
5286 021026 005204          45:   INC    R4            ;; DON'T SUPPRESS ANYMORE 0'S
5287 021030 052703 000060   BIS    #'0,R3        ;; MAKE THIS DIGIT ASCII
5288 021034 052703 000040   55:   BIS    #' ,R3      ;; MAKE ASCII IF NOT ALREADY
5289 021040 110337 021104   MOVB   R3,R5          ;; SAVE FOR TYPING
5290 021044 104401 021104   TYPE   ,R5            ;; GO TYPE THIS DIGIT
5291 021050 105337 021106   75:   DECB   $OCNT        ;; COUNT BY 1
5292 021054 003347          BGT    25            ;; BR IF MORE TO DO
5293 021056 002402          BLT    65            ;; BR IF DONE
5294 021060 005204          INC    R4            ;; INSURE LAST DIGIT ISN'T A BLANK
5295 021062 000744          BR     25            ;; GO DO THE LAST DIGIT
5296 021064 012605          65:   MOV    (SP)+,R5      ;; RESTORE R5
5297 021066 012604          MOV    (SP)+,R4      ;; RESTORE R4
5298 021070 012603          MOV    (SP)+,R3      ;; RESTORE R3
5299 021072 016666 000002 000004  MOV    2(SP),4(SP)    ;; SET THE STACK FOR RETURNING
5300 021100 012616          MOV    (SP)+,(SP)
5301 021102 000002          RTI                    ;; RETURN
5302 021104 000          85:   .BYTE  0            ;; STORAGE FOR ASCII DIGIT
5303 021105 000          .BYTE  0            ;; TERMINATOR FOR TYPE ROUTINE
5304 021106 000          $OCNT: .BYTE  0        ;; OCTAL DIGIT COUNTER
5305 021107 000          $OFILL: .BYTE  0     ;; ZERO FILL SWITCH
5306 021110 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 $TYPDS:
5321 021112 010046          MOV    R0,-(SP)      ;; PUSH R0 ON STACK
5322 021114 010146          MOV    R1,-(SP)      ;; PUSH R1 ON STACK
5323 021116 010246          MOV    R2,-(SP)      ;; PUSH R2 ON STACK
5324 021120 010346          MOV    R3,-(SP)      ;; PUSH R3 ON STACK
5325 021122 010546          MOV    R5,-(SP)      ;; PUSH R5 ON STACK
5326 021124 012746 020200   MOV    #20200,-(SP)  ;; SET BLANK SWITCH AND SIGN
5327 021130 016605 000020   MOV    20(SP),R5     ;; GET THE INPUT NUMBER
5328 021134 100004          BPL    15            ;; BR IF INPUT IS POS.
5329 021136 005405          NEG    R5            ;; MAKE THE BINARY NUMBER POS.
5330 021140 112766 000055 000001  MOVB   #'-,1(SP)     ;; MAKE THE ASCII NUMBER NEG.
5331 021146 005000          15:   CLR    R0            ;; ZERO THE CONSTANTS INDEX
    
```



```

5332 021150 012703 021326      MOV      #SDBLK,R3      ;; SETUP THE OUTPUT POINTER
5333 021154 112723 000040      MOVB     #' ,(R3)+     ;; SET THE FIRST CHARACTER TO A BLANK
5334 021160 005002           25:     CLR      R2          ;; CLEAR THE BCD NUMBER
5335 021162 016001 021316      MOV      $DTBL(RO),R1  ;; GET THE CONSTANT
5336 021166 160105           35:     SUB      R1,R5        ;; FORM THIS BCD DIGIT
5337 021170 002402           BLT      45           ;; BR IF DONE
5338 021172 005202           INC      R2          ;; INCREASE THE BCD DIGIT BY 1
5339 021174 000774           BR       35
5340 021176 060105           45:     ADD      R1,R5        ;; ADD BACK THE CONSTANT
5341 021200 005702           TST      R2          ;; CHECK IF BCD DIGIT=0
5342 021202 001002           BNE      55          ;; FALL THROUGH IF 0
5343 021204 105716           TSTB    (SP)         ;; STILL DOING LEADING 0'S?
5344 021206 100407           BMI      75          ;; BR IF YES
5345 021210 106316           55:     ASLB    (SP)        ;; MSD?
5346 021212 103003           BCC      65          ;; BR IF NO
5347 021214 116663 000001 177777      MOVB     1(SP),-1(R3)  ;; YES--SET THE SIGN
5348 021222 052702 000060           65:     BIS      #'0,R2     ;; MAKE THE BCD DIGIT ASCII
5349 021226 052702 000040           75:     BIS      #' ,R2     ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
5350 021232 110223           MOVB     R2,(R3)+     ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
5351 021234 005720           TST      (RO)+       ;; JUST INCREMENTING
5352 021236 020027 000010           CMP      RO,#10      ;; CHECK THE TABLE INDEX
5353 021242 002746           BLT      25          ;; GO DO THE NEXT DIGIT
5354 021244 003002           BGT      85          ;; GO TO EXIT
5355 021246 010502           MOV      R5,R2       ;; GET THE LSD
5356 021250 000764           BR       65          ;; GO CHANGE TO ASCII
5357 021252 105726           85:     TSTB    (SP)+       ;; WAS THE LSD THE FIRST NON-ZERO?
5358 021254 100003           BPL      95          ;; BR IF NO
5359 021256 116663 177777 177776      MOVB     -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING
5360 021264 105013           95:     CLRB    (R3)        ;; SET THE TERMINATOR
5361 021266 012605           MOV      (SP)+,R5     ;; POP STACK INTO R5
5362 021270 012603           MOV      (SP)+,R3     ;; POP STACK INTO R3
5363 021272 012602           MOV      (SP)+,R2     ;; POP STACK INTO R2
5364 021274 012601           MOV      (SP)+,R1     ;; POP STACK INTO R1
5365 021276 012600           MOV      (SP)+,R0     ;; POP STACK INTO R0
5366 021300 104401 021326      TYPE     ,SDBLK       ;; NOW TYPE THE NUMBER
5367 021304 016666 000002 000004      MOV      2(SP),4(SP)  ;; ADJUST THE STACK
5368 021312 012616           MOV      (SP)+,(SP)
5369 021314 000002           RTI                    ;; RETURN TO USER
5370 021316 023420           $DTBL:  10000.
5371 021320 001750           1000.
5372 021322 000144           100.
5373 021324 000012           10.
5374 021326 000004           $SDBLK: .BLKW 4
5375
5376           .SBTTL  TTY INPUT ROUTINE
5377
5378           ;; *****
5379           .ENABL  LSB
5380 021336 000000      $TKCNT: .WORD 0      ;; NUMBER OF ITEMS IN QUEUE
5381 021340 000000      $TKQIN: .WORD 0     ;; INPUT POINTER
5382 021342 000000      $TKQOUT: .WORD 0    ;; OUTPUT POINTER
5383 021344 000002      $TKQSRT: .BLKB 2    ;; TTY KEYBOARD QUEUE
5384           $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,STKINT
5392 ;# RETURN
5393 ;
5394 STKINT: CLR STKCNT ;: CLEAR COUNT OF ITEMS IN QUEUE
5395 MOV #STKQSR,STKQIN ;: MOVE THE STARTING ADDRESS OF THE
5396 MOV STKQIN,STKQOUT ;: QUEUE INTO THE INPUT & OUTPUT POINTERS.
5397 MOV #STKSRV,@STKVEC ;: INITIALIZE THE KEYBOARD VECTOR
5398 MOV #200,@STKVEC+2 ;: "BR" LEVEL 4
5399 TST @STKB ;: CLEAR DONE FLAG
5400 MOV #100,@STKS ;: ENABLE TTY KEYBOARD INTERRUPT
5401 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 ) STKINT IS CALLED AND
5408 ;#UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
5409 ;
5410 STKSRV: MOVB @STKB,-(SP) ;: PICKUP THE CHARACTER
5411 BIC #C177,(SP) ;: STRIP THE JUNK
5412 CMP (SP),#3 ;: IS IT A CONTROL C?
5413 BNE 15 ;: BRANCH IF NO
5414 TYPE ,SCNTLC ;: TYPE A CONTROL-C ( C )
5415 JSR PC,STKINT ;: INIT THE KEYBOARD
5416 TST (SP)+ ;: CLEAN UP STACK
5417 JMP START2 ;: CONTROL C RESTART
5418 15: CMP (SP),#7 ;: IS IT A CONTROL G?
5419 BNE 25 ;: BRANCH IF NO
5420 CMP #SWREG,SWR ;: IS SOFT-SWR SELECTED?
5421 BEQ 65 ;: GO TO SWR CHANGE
5422
5423 25:
5424 CMP #2,STKCNT ;: IS THE QUEUE FULL?
5425 BNE 35 ;: BRANCH IF NO
5426 TYPE ,SBELL ;: RING THE TTY BELL
5427 TST (SP)+ ;: CLEAN CHARACTER OFF OF STACK
5428 BR 55 ;: EXIT
5429 35: CMP (SP),#23 ;: IS IT A CONTROL-S?
5430 BNE 325 ;: BRANCH IF NO
5431 CLR @STKS ;: DISABLE TTY KEYBOARD INTERRUPTS
5432 TST (SP)+ ;: CLEAN CHAR OFF STACK
5433 315: TSTB @STKS ;: WAIT FOR A CHAR
5434 BPL 315 ;: LOOP UNTIL ITS THERE
5435 MOVB @STKB,-(SP) ;: GET THE CHARACTER
5436 BIC #C177,(SP) ;: MAKE IT 7-BIT ASCII
5437 CMP (SP)+,#21 ;: IS IT A CONTROL-Q?
5438 BNE 315 ;: BRANCH IF NO
5439 MOV #100,@STKS ;: REENABLE TTY KEYBOARD INTERRUPTS
5440 RTI ;: RETURN
5441 325: INC STKCNT ;: COUNT THIS CHARACTER
5442 CMP (SP),#140 ;: IS IT UPPER CASE?
5443 BLT 45 ;: BRANCH IF YES

```



```

5444 021572 021627 000175          CMP      (SP),#175      ;; IS IT A SPECIAL CHAR?
5445 021576 003002          BGT      45             ;; BRANCH IF YES
5446 021600 042716 000040          BIC      #40,(SP)      ;; MAKE IT UPPER CASE
5447 021604 112677 177530          45:     MOVB     (SP)+,@STKQIN ;; AND PUT IT IN QUEUE
5448 021610 005237 021340          INC      STKQIN        ;; UPDATE THE POINTER
5449 021614 023727 021340 021346     CMP      STKQIN,@STKQEND ;; GO OFF THE END?
5450 021622 001003          BNE      55           ;; BRANCH IF NO
5451 021624 012737 021344 021340     MOV      @STKQSRT,STKQIN ;; RESET THE POINTER
5452 021632 000002          55:     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 021634 022737 000176 001140     SCKSWR: CMP      @SWREG,SWR  ;; IS THE SOFT-SWR SELECTED
5460 021642 001124          BNE      155          ;; EXIT IF NOT
5461 021644 105777 157274          TSTB     @STKS        ;; IS A CHAR WAITING?
5462 021650 100121          BPL      155          ;; IF NOT, EXIT
5463 021652 117746 157270          MOVB     @STKB,-(SP)  ;; YES
5464 021656 042716 177600          BIC      # (177,(SP)  ;; MAKE IT 7-BIT ASCII
5465 021662 021627 000007          CMP      (SP),#7     ;; IS IT A CONTROL-G?
5466 021666 001300          BNE      25           ;; IF NOT, PUT IT IN THE TTY QUEUE
5467
5468          ;; AND EXIT
5469
5470          ;; *****
5471          ;; CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
5472          ;; ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
5473          ;; CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
5474 021670 123727 001134 000001     65:     CMPB     $AUTOB,#1   ;; ARE WE RUNNING IN AUTO-MODE?
5475 021676 001674          BEQ      25           ;; BRANCH IF YES
5476 021700 005726          TST      (SP)+       ;; CLEAR CONTROL-G OFF STACK
5477 021702 004737 021346          JSR      PC,STKINT   ;; FLUSH THE TTY INPUT QUEUE
5478 021706 005077 157232          CLR      @STKS      ;; DISABLE TTY KEYBOARD INTERRUPTS
5479 021712 112737 000001 001135     MOVB     #1,$INTAG   ;; SET INTERRUPT MODE INDICATOR
5480 021720 104401 022560          SGTSWR: TYPE     , $CNTLG  ;; ECHO THE CONTROL-G ( G)
5481 021724 104401 022565          TYPE     , $MSWR     ;; TYPE CURRENT CONTENTS
5482 021730 013746 000176          MOV      SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
5483 021734 104402          TYPOC    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
5484 021736 104401 022576          TYPE     , $MNEW     ;; PROMPT FOR NEW SWR
5485 021742 005046          195:    CLR      -(SP)   ;; CLEAR COUNTER
5486 021744 005046          CLR      -(SP)   ;; THE NEW SWR
5487 021746 105777 157172          75:     TSTB     @STKS      ;; CHAR THERE?
5488 021752 100375          BPL      75        ;; IF NOT TRY AGAIN
5489
5490 021754 117746 157166          MOVB     @STKB,-(SP) ;; PICK UP CHAR
5491 021760 042716 177600          BIC      # (177,(SP) ;; MAKE IT 7-BIT ASCII
5492
5493 021764 021627 000003          CMP      (SP),#3     ;; IS IT A CONTROL-C?
5494 021770 001015          BNE      95           ;; BRANCH IF NOT
5495 021772 104401 022546          TYPE     , $CNTLC    ;; YES, ECHO CONTROL-C ( C)
5496 021776 062706 000006          ADD      #6,SP       ;; CLEAN UP STACK
5497 022002 123727 001135 000001     CMPB     $INTAG,#1   ;; REENABLE TTY KEYBOARD INTERRUPTS?
5498 022010 001003          BNE      85           ;; BRANCH IF NO
5499 022012 012777 000100 157124     MOV      #100,@STKS  ;; ALLOW TTY KEYBOARD INTERRUPTS

```



```

5500 022020 000137 004660      85:  JMP      START2      ;; CONTROL-C RESTART
5501
5502
5503 022024 021627 000025      95:  CMP      (SP),#25    ;; IS IT A CONTROL-U?
5504 022030 001005                BNE      105          ;; BRANCH IF NOT
5505 022032 104401 022553        TYPE    ,%CNTLU      ;; YES, ECHO CONTROL-U ( U)
5506 022036 062706 000006      205:  ADD      #6,SP        ;; IGNORE PREVIOUS INPUT
5507 022042 000737                BR       195          ;; LET'S TRY IT AGAIN
5508
5509
5510 022044 021627 000015      105:  CMP      (SP),#15    ;; IS IT A <CR>?
5511 022050 001022                BNE      165          ;; BRANCH IF NO
5512 022052 005766 000004        TST     4(SP)        ;; YES, IS IT THE FIRST CHAR?
5513 022056 001403                BEQ     115          ;; BRANCH IF YES
5514 022060 016677 000002 157052    MOV     2(SP),%SWR    ;; SAVE NEW SWR
5515 022066 062706 000006      115:  ADD      #6,SP        ;; CLEAR UP STACK
5516 022072 104401 001215      145:  TYPE    ,%CRLF       ;; ECHO <CR> AND <LF>
5517 022076 123727 001135 000001    CMPB   %INTAG,#1     ;; RE-ENABLE TTY KBD INTERRUPTS?
5518 022104 001003                BNE     155          ;; BRANCH IF NOT
5519 022106 012777 000100 157030    MOV     #100,%STKS   ;; RE-ENABLE TTY KBD INTERRUPTS
5520 022114 000002      155:  RTI                    ;; RETURN
5521 022116 004737 020614      165:  JSR     PC,%STPEC    ;; ECHO CHAR
5522 022122 021627 000060        CMP     (SP),#60     ;; CHAR < 0?
5523 022126 002420                BLT     185          ;; BRANCH IF YES
5524 022130 021627 000067        CMP     (SP),#67     ;; CHAR > 7?
5525 022134 003015                BGT     185          ;; BRANCH IF YES
5526 022136 042726 000060        BIC     #60,(SP)+    ;; STRIP-OFF ASCII
5527 022142 005766 000002        TST     2(SP)        ;; IS THIS THE FIRST CHAR
5528 022146 001403                BEQ     175          ;; BRANCH IF YES
5529 022150 006316                ASL     (SP)         ;; NO, SHIFT PRESENT
5530 022152 006316                ASL     (SP)         ;; CHAR OVER TO MAKE
5531 022154 006316                ASL     (SP)         ;; ROOM FOR NEW ONE.
5532 022156 005266 000002      175:  INC     2(SP)        ;; KEEP COUNT OF CHAR
5533 022162 056616 177776        BIS     -2(SP),(SP)  ;; SET IN NEW CHAR
5534 022166 000667                BR      75           ;; GET THE NEXT ONE
5535 022170 104401 001214      185:  TYPE    ,%QUES       ;; TYPE ?<CR><LF>
5536 022174 000720                BR      205          ;; 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 022176 011646 000004 000002  $RDCHR: MOV     (SP),-(SP)    ;; PUSH DOWN THE PC AND
5549 022200 016666 000004                MOV     4(SP),2(SP)  ;; THE PS
5550 022206 005066 000004                CLR     4(SP)        ;; GET READY FOR A CHARACTER
5551 022212 005046                CLR     -(SP)        ;; PUT NEW PS ON STACK
5552 022214 012746 022222                MOV     #645,-(SP)  ;; PUT NEW PC ON STACK
5553 022220 000002                RTI                    ;; POP NEW PC AND PS
5554 022222
5555 022222 005737 021336      15:  TST     $STKCNT      ;; WAIT ON A CHARACTER

```



```

5556 022226 001775          BEQ      15
5557 022230 005337 021336    DEC      $TKCNT          ;; DECREMENT THE COUNTER
5558 022234 117766 177102 000004  MOVB    $STKQOUT,4(SP) ;; GET ONE CHARACTER
5559 022242 005237 021342          INC      $TKQOUT        ;; UPDATE THE POINTER
5560 022246 023727 021342 021346  CMP     $STKQOUT,$STKQEND ;; DID IT GO OFF OF THE END?
5561 022254 001003          BNE     25              ;; BRANCH IF NO
5562 022256 012737 021344 021342  MOV     $STKQSR, $TKQOUT ;; RESET THE POINTER
5563 022264 000002          25:    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 022266 010346          SRDLIN: MOV     R3,-(SP)      ;; SAVE R3
5572 022270 005046          CLR     -(SP)          ;; CLEAR THE RUBOUT KEY
5573 022272 012703 022522    15:    MOV     $TTYIN,R3    ;; GET ADDRESS
5574 022276 022703 022546    25:    CMP     $TTYIN+20.,R3  ;; BUFFER FULL?
5575 022302 101456          BLOS   45              ;; BR IF YES
5576 022304 104410          RDCHR          ;; GO READ ONE CHARACTER FROM THE TTY
5577 022306 112613          MOVB   (SP)+,(R3)      ;; GET CHARACTER
5578 022310 122713 000177    105:   CMPB   #177,(R3)       ;; IS IT A RUBOUT
5579 022314 001022          BNE   55              ;; BR IF NO
5580 022316 005716          TST   (SP)            ;; IS THIS THE FIRST RUBOUT?
5581 022320 001007          BNE   65              ;; BR IF NO
5582 022322 112737 000134 022520  MOVB   #' ,95          ;; TYPE A BACK SLASH
5583 022330 104401 022520    TYPE   ,95
5584 022334 012716 177777    MOV    #-1,(SP)        ;; SET THE RUBOUT KEY
5585 022340 005303          65:    DEC     R3              ;; BACKUP BY ONE
5586 022342 020327 022522    CMP    R3,$TTYIN       ;; STACK EMPTY?
5587 022346 103434          BLO   45              ;; BR IF YES
5588 022350 111337 022520    MOVB   (R3),95         ;; SETUP TO TYPEOUT THE DELETED CHAR.
5589 022354 104401 022520    TYPE   ,95
5590 022360 000746          BR     25              ;; GO READ ANOTHER CHAR.
5591 022362 005716          55:    TST   (SP)            ;; RUBOUT KEY SET?
5592 022364 001406          BEQ   75              ;; BR IF NO
5593 022366 112737 000134 022520  MOVB   #' ,95          ;; TYPE A BACK SLASH
5594 022374 104401 022520    TYPE   ,95
5595 022400 005016          CLR   (SP)            ;; CLEAR THE RUBOUT KEY
5596 022402 122713 000025    75:    CMPB   #25,(R3)        ;; IS CHARACTER A CTRL U?
5597 022406 001003          BNE   85              ;; BR IF NO
5598 022410 104401 022553    TYPE   ,%CNTLU        ;; TYPE A CONTROL "U"
5599 022414 000726          BR     15              ;; GO START OVER
5600 022416 122713 000022    85:    CMPB   #22,(R3)        ;; IS CHARACTER A " R"?
5601 022422 001011          BNE   35              ;; BRANCH IF NO
5602 022424 105013          CLRB  (R3)            ;; CLEAR THE CHARACTER
5603 022426 104401 001215    TYPE   ,%CRLF         ;; TYPE A "CR" & "LF"
5604 022432 104401 022522    TYPE   , $TTYIN        ;; TYPE THE INPUT STRING
5605 022436 000717          BR     25              ;; GO PICKUP ANOTHER CHARACTER
5606 022440 104401 001214    45:    TYPE   ,%QUES         ;; TYPE A '?'
5607 022444 000712          BR     15              ;; CLEAR THE BUFFER AND LOOP
5608 022446 111337 022520    35:    MOVB   (R3),95         ;; ECHO THE CHARACTER
5609 022452 104401 022520    TYPE   ,95
5610 022456 122723 000015    CMPB   #15,(R3)+      ;; CHECK FOR RETURN
5611 022462 001305          BNE   25              ;; LOOP IF NOT RETURN
    
```



```

5612 022464 105063 177777          CLRB    -1(R3)          ;; CLEAR RETURN (THE 15)
5613 022470 104401 001216          TYPE    , $LF          ;; TYPE A LINE FEED
5614 022474 005726                   TST     (SP)+          ;; CLEAN RUBOUT KEY FROM THE STACK
5615 022476 012603                   MOV     (SP)+, R3      ;; RESTORE R3
5616 022500 011646                   MOV     (SP), -(SP)    ;; ADJUST THE STACK AND PUT ADDRESS OF THE
5617 022502 016666 000004 000002    MOV     4(SP), 2(SP)   ;; FIRST ASCII CHARACTER ON IT
5618 022510 012766 022522 000004    MOV     @STTYIN, 4(SP)
5619 022516 000002                   RTI                    ;; RETURN
5620 022520      000          95:    . BYTE  0          ;; STORAGE FOR ASCII CHAR. TO TYPE
5621 022521      000          . BYTE  0          ;; TERMINATOR
5622 022522 000024          $TTYIN: . BLKB 20.    ;; RESERVE 20. BYTES FOR TTY INPUT
5623 022546 041536 005015      000    $CNTLC: . ASCIIZ / C/<15><12>  ;; CONTROL "C"
5624 022553      136 006525 000012    $CNTLU: . ASCIIZ / U/<15><12>  ;; CONTROL "U"
5625 022560 043536 005015      000    $CNTLG: . ASCIIZ / G/<15><12>  ;; CONTROL "G"
5626 022565      015 051412 051127    $MSWR:  . ASCIIZ <15><12>/SWR = /
5627 022572 036440 000040          $MNEW:  . ASCIIZ / NEW = /
5628 022576 020040 042516 020127
5629 022604 020075      000
5630          022610          . EVEN
5631
5632          . SBTTL SCOPE HANDLER ROUTINE
5633
5634          ;; *****
5635          ;; *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
5636          ;; *AND LOAD THE TEST NUMBER($STNM) 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=10T
5644
5645          $SCOPE:
5646 022610 104407          CKSWR          ;; TEST FOR CHANGE IN SOFT-SWR
5647 022612 032777 040000 156320 15:    BIT     #BIT14, @SWR    ;; LOOP ON PRESENT TEST?
5648 022620 001101          BNE     $OVER        ;; YES IF SW14=1
5649          ;; #####START OF CODE FOR THE XOR TESTER#####
5650 022622 000416          $XTSTR: BR     65     ;; IF RUNNING ON THE "XOR" TESTER CHANGE
5651          ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
5652 022624 013746 000004          MOV     @ERRVEC, -(SP) ;; SAVE THE CONTENTS OF THE ERROR VECTOR
5653 022630 012737 022650 000004    MOV     #55, @ERRVEC   ;; SET FOR TIMEOUT
5654 022636 005737 177060          TST     @177060        ;; TIME OUT ON XOR?
5655 022642 012637 000004          MOV     (SP)+, @ERRVEC ;; RESTORE THE ERROR VECTOR
5656 022646 000453          BR     $SVLAD         ;; GO TO THE NEXT TEST
5657 022650 022626          55:    CMP     (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
5658 022652 012637 000004          MOV     (SP)+, @ERRVEC ;; RESTORE THE ERROR VECTOR
5659 022656 000413          BR     75            ;; LOOP ON THE PRESENT TEST
5660 022660          55:    ;; #####END OF CODE FOR THE XOR TESTER#####
5661 022660 105737 001103          25:    TSTB    $ERFLG    ;; HAS AN ERROR OCCURRED?
5662 022664 001421          BEQ     35            ;; BR IF NO
5663 022666 123737 001115 001103    CMPB    $ERMAX, $ERFLG ;; MAX. ERRORS FOR THIS TEST OCCURRED?
5664 022674 101015          BHI     35            ;; BR IF NO
5665 022676 032777 001000 156234    BIT     #BIT09, @SWR    ;; LOOP ON ERROR?
5666 022704 001404          BEQ     45            ;; BR IF NO
5667 022706 013737 001110 001106 75:    MOV     $LPERR, $LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
    
```



```

5668 022714 000443          BR      $OVER
5669 022716 105037 001103 45:    CLR    $ERFLG      ;; ZERO THE ERROR FLAG
5670 022722 005037 001204          CLR    $TIMES      ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
5671 022726 000415          BR      15          ;; ESCAPE TO THE NEXT TEST
5672 022730 032777 004000 156202 35:    BIT    $BIT11,$SWR  ;; INHIBIT ITERATIONS?
5673 022736 001011          BNE    15          ;; BR IF YES
5674 022740 005737 001100          TST    $PASS       ;; IF FIRST PASS OF PROGRAM
5675 022744 001406          BEQ    15          ;; INHIBIT ITERATIONS
5676 022746 005237 001104          INC    $ICNT       ;; INCREMENT ITERATION COUNT
5677 022752 023737 001204 001104          CMP    $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
5678 022760 002021          BGE    $OVER       ;; BR IF MORE ITERATION REQUIRED
5679 022762 012737 000001 001104 15:    MOV    $1,$ICNT    ;; REINITIALIZE THE ITERATION COUNTER
5680 022770 013737 023040 001204          MOV    $SMXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
5681 022776 105237 001102          $SVLAD: INCB   $STSTNM ;; COUNT TEST NUMBERS
5682 023002 011637 001106          MOV    ($SP),$SLPADR ;; SAVE SCOPE LOOP ADDRESS
5683 023006 011637 001110          MOV    ($SP),$SLPERR ;; SAVE ERROR LOOP ADDRESS
5684 023012 005037 001206          CLR    $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
5685 023016 112737 000001 001115          MOVB   $1,$SERMAX  ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
5686 023024 013777 001102 156110 $OVER: MOV    $STSTNM,$DISPLAY ;; DISPLAY TEST NUMBER
5687 023032 013716 001106          MOV    $SLPADR,$SP ;; FUDGE RETURN ADDRESS
5688 023036 000002          RTI              ;; FIXES PS
5689 023040 000001          $SMXCNT: 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          $SAVREG:
5709 023042 010046          MOV    R0,-($SP)   ;; PUSH R0 ON STACK
5710 023044 010146          MOV    R1,-($SP)   ;; PUSH R1 ON STACK
5711 023046 010246          MOV    R2,-($SP)   ;; PUSH R2 ON STACK
5712 023050 010346          MOV    R3,-($SP)   ;; PUSH R3 ON STACK
5713 023052 010446          MOV    R4,-($SP)   ;; PUSH R4 ON STACK
5714 023054 010546          MOV    R5,-($SP)   ;; PUSH R5 ON STACK
5715 023056 016646 000022          MOV    22($SP),-($SP) ;; SAVE PS OF MAIN FLOW
5716 023062 016646 000022          MOV    22($SP),-($SP) ;; SAVE PC OF MAIN FLOW
5717 023066 016646 000022          MOV    22($SP),-($SP) ;; SAVE PS OF CALL
5718 023072 016646 000022          MOV    22($SP),-($SP) ;; SAVE PC OF CALL
5719 023076 000002          RTI
5720
5721          ;*RESTORE R0-R5
5722          ;*CALL:
5723          ;*   RESREG
    
```

5724 023100  
 5725 023100 012666 000022  
 5726 023104 012666 000022  
 5727 023110 012666 000022  
 5728 023114 012666 000022  
 5729 023120 012605  
 5730 023122 012604  
 5731 023124 012603  
 5732 023126 012602  
 5733 023130 012601  
 5734 023132 012600  
 5735 023134 000002  
 5736  
 5737  
 5738  
 5739  
 5740  
 5741  
 5742  
 5743  
 5744  
 5745 023136 010046  
 5746 023140 016600 000002  
 5747 023144 005740  
 5748 023146 111000  
 5749 023150 006300  
 5750 023152 016000 023172  
 5751 023156 000200  
 5752  
 5753  
 5754  
 5755  
 5756 023160 011646  
 5757 023162 016666 000004 000002  
 5758 023170 000002  
 5759  
 5760  
 5761  
 5762  
 5763  
 5764  
 5765  
 5766  
 5767 023172 023160  
 5768 023174 020444  
 5769 023176 020710  
 5770 023200 020664  
 5771 023202 020724  
 5772 023204 021112  
 5773  
 5774 023206 021724  
 5775  
 5776 023210 021634  
 5777 023212 022176  
 5778 023214 022266  
 5779 023216 023042

```

$RESREG:
MOV (SP)+, 22(SP) ;; RESTORE PC OF CALL
MOV (SP)+, 22(SP) ;; RESTORE PS OF CALL
MOV (SP)+, 22(SP) ;; RESTORE PC OF MAIN FLOW
MOV (SP)+, 22(SP) ;; RESTORE PS OF MAIN FLOW
MOV (SP)+, R5 ;; POP STACK INTO R5
MOV (SP)+, R4 ;; POP STACK INTO R4
MOV (SP)+, R3 ;; POP STACK INTO R3
MOV (SP)+, R2 ;; POP STACK INTO R2
MOV (SP)+, R1 ;; POP STACK INTO R1
MOV (SP)+, R0 ;; POP STACK INTO R0
RTI
    
```

.SBTTL TRAP DECODER

```

;*****
;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
;GO TO THAT ROUTINE.
    
```

```

$TRAP: MOV R0, -(SP) ;; SAVE R0
MOV 2(SP), R0 ;; GET TRAP ADDRESS
TST -(R0) ;; BACKUP BY 2
MOVB (R0), R0 ;; GET RIGHT BYTE OF TRAP
ASL R0 ;; POSITION FOR INDEXING
MOV $TRPAD(R0), R0 ;; INDEX TO TABLE
RTS R0 ;; GO TO ROUTINE
    
```

;; THIS IS USE TO HANDLE THE "GETPRI" MACRO

```

$TRAP2: MOV (SP), -(SP) ;; MOVE THE PC DOWN
MOV 4(SP), 2(SP) ;; MOVE THE PSW DOWN
RTI ;; RESTORE THE PSW
    
```

.SBTTL TRAP TABLE

```

;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;BY THE "TRAP" INSTRUCTION.
    
```

	ROUTINE	
\$TRPAD:	WORD \$TRAP2	
	\$TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
	\$TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
	\$GTSWR	TRAP+6(104406) GET SOFT-SWR SETTING
	\$CKSWR	TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
	\$RDCHR	TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
	\$RDLIN	TRAP+11(104411) TTY TYPEIN STRING ROUTINE
	\$SAVREG	TRAP+12(104412) SAVE RO-R5 ROUTINE



```

5780 023220 023100          SRESREG ;;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 BINARY NUMBER TO AN
5786          ;;UNSIGNED DECIMAL ASCIZ NUMBER.
5787          ;;CALL
5788          ;;      MOV      NUMBER,-(SP)      ;;PUT BINARY NUMBER ON THE STACK
5789          ;;      JSR      PC,@#SSB2D        ;;CALL
5790          ;;      RETURN                      ;;ADDRESS OF THE 1ST ASCIZ CHAR. IS ON THE STACK
5791
5792
5793 023222 016637 000002 023252  SSB2D:  MOV      2(SP),15      ;;SAVE BINARY NUMBER
5794 023230 012746 023252          MOV      #15,-(SP)    ;;SET POINTER
5795 023234 004737 023256          JSR      PC,@#SDB2D   ;;CALL DOUBLE LENGTH CONVERT
5796 023240 062716 000005          ADD      #5,(SP)     ;;ONLY ALLOW FIVE CHARACTERS
5797 023244 012666 000002          MOV      (SP)+,2(SP) ;;PICKUP POINTER
5798 023250 000207          RTS      PC         ;;RETURN
5799 023252 000000 000000          15:     .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,@#SDB2D        ;;CALL
5810          ;;      RETURN                      ;;THE FIRST ADDRESS OF ASCIZ
5811          ;;                                  ;;IS ON THE STACK
5812
5813
5814          SDB2D:  SAVREG                      ;;SAVE REGISTERS
5815 023256 104412          MOV      2(SP),R2    ;;PICKUP THE DATA POINTER
5816 023260 016602 000002          MOV      #SDECVL,R0  ;;GET ADDRESS OF "SDECVL" STRING
5817 023264 012700 023436          MOV      R0,2(SP)    ;;PUT ADDRESS OF ASCIZ STRING ON STACK
5818 023270 010066 000002          MOV      (R2)+,R1    ;;PICKUP THE BINARY NUMBER
5819 023274 012201          MOV      (R2)+,R2
5820 023300 012737 000012 023354          MOV      #10,45      ;;SET UP TO DO 10 CONVERSIONS
5821 023306 012704 023366          MOV      #STNPNR,R4   ;;ADDRESS OF TEN POWER
5822 023312 012705 023370          MOV      #STNPNR+2,R5
5823 023316 005003          15:     CLR      R3         ;;CLEAR PARTIAL
5824 023320 161401          25:     SUB      (R4),R1      ;;SUBTRACT TEN POWER
5825 023322 005602          SBC      R2
5826 023324 161502          SUB      (R5),R2
5827 023326 002402          BLT      35           ;;BR IF TEN POWER TO LARGE
5828 023330 005203          INC      R3         ;;ADD 1 TO PARTIAL
5829 023332 000772          BR      25         ;;LOOP
5830 023334 062401          35:     ADD      (R4)+,R1    ;;RESTORE SUBTRACTED VALUE
5831 023336 005502          ADC      R2
5832 023340 062402          ADD      (R4)+,R2
5833 023342 022525          CMP      (R5)+,(R5)+ ;;MOVE TO NEXT TEN POWER
5834 023344 052703 000060          BIS      #'0,R3     ;;CHANGE PARTIAL TO ASCII
5835 023350 110320          MOVB    R3,(R0)+    ;;SAVE IT
    
```

```

5836 023352 005327          DEC      (PC)+      ;; DONE?
5837 023354 000000          4$: .WORD      0
5838 023356 001357          BNE      1$      ;; BR IF NO
5839 023360 105020          CLRB     (RO)+    ;; TERMINATOR
5840 023362 104413          RESREG                    ;; RESTORE REGISTERS
5841 023364 000207          RTS      PC      ;; RETURN
5842 023366 145000          $TNPHR: 145000    ;; 1.0E09
5843 023370 035632          35632
5844 023372 160400          160400          ;; 1.0E08
5845 023374 002765          2765
5846 023376 113200          113200          ;; 1.0E07
5847 023400 000230          230
5848 023402 041100          041100          ;; 1.0E06
5849 023404 000017          17
5850 023406 103240          103240          ;; 1.0E05
5851 023410 000001          1
5852 023412 023420          23420           ;; 1.0E04
5853 023414 000000          0
5854 023416 001750          1750           ;; 1.0E03
5855 023420 000000          0
5856 023422 000144          144           ;; 1.0E02
5857 023424 000000          0
5858 023426 000012          12           ;; 1.0E01
5859 023430 000000          0
5860 023432 000001          1           ;; 1.0E00
5861 023434 000000          0
5862 023436 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          $$SUPRS: MOV      RO, -(SP)      ;; SAVE RO
5874          023452 010046          MOV      4(SP), RO    ;; PICKUP THE POINTER
5875          023454 016600 000004          1$: TSTB     (RO)      ;; TERMINATOR?
5876          023460 105710          BEQ      2$          ;; BR IF YES
5877          023462 001403          CMPB     #'0, (RO)+   ;; IS THIS AN ASCII "0" ?
5878          023464 122720 000060          BEQ      1$          ;; BR IF YES
5879          023470 001773          DEC      RO          ;; BACKUP BY "1"
5880          023472 005300          MOV      RO, 3$      ;; SAVE FOR TYPING
5881          023474 010037 023502          TYPE                    ;; GO TYPE
5882          023500 104401          .WORD    0           ;; ASCIZ POINTER GOES HERE
5883          023502 000000          MOV      (SP)+, RO    ;; RESTORE RO
5884          023504 012600          MOV      (SP)+, (SP)  ;; RESTORE THE STACK
5885          023506 012616          RTS      PC          ;; RETURN
5886          023510 000207
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,SRAND ;*CALL THE ROUTINE
5895 ;* RETURN ;*RETURN HERE THE RANDOM
5896 ;* ;*NUMBER WILL BE IN
5897 ;* ;*SHINUM,SLONUM
5898
5899 SRAND:
5900 MOV RO,-(SP) ;*PUSH RO ON STACK
5901 MOV R1,-(SP) ;*PUSH R1 ON STACK
5902 MOV R2,-(SP) ;*PUSH R2 ON STACK
5903 MOV $LONUM,R0 ;*SET RO WITH LOW
5904 MOV $SHINUM,R1 ;*SET R1 WITH HIGH
5905 MOV #7,R2 ;*SET SHIFT COUNT
5906 1$: ASL R0 ;*SHIFT RO LEFT AND
5907 ROL R1 ;*ROTATE CARRY INTO R1 AND
5908 INC R2 ;*CHECK FOR DONE
5909 BNE 1$ ;*CONTINUE SHIFT LOOP
5910 ADD $LONUM,R0 ;*ADD NUMBER TO MAKE X 129
5911 ADC R1 ;*PROPOGATE CARRY
5912 ADD $SHINUM,R1 ;*ADD NUMBER TO MAKE X 129
5913 ADD #1057,R0 ;*ADD LOW CONSTANT
5914 ADC R1 ;*PROPOGATE CARRY
5915 ADD #47401,R1 ;*ADD HIGH CONSTANT
5916 MOV RO,$LONUM ;*SAVE RO
5917 MOV R1,$SHINUM ;*SAVE R1
5918 MOV (SP)+,R2 ;*POP STACK INTO R2
5919 MOV (SP)+,R1 ;*POP STACK INTO R1
5920 MOV (SP)+,R0 ;*POP STACK INTO RO
5921 RTS PC ;*RETURN
5922 $SHINUM: .WORD 176543
5923 $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
    
```

5948				:*	+2	QUOTIENT	ALL ZEROS	ALL ONES
5949								
5950	023614							
5951	023614	104400						
5952	023616	042716	000017					
5953	023622	010046						
5954	023624	010146						
5955	023626	010246						
5956	023630	010346						
5957	023632	005046						
5958	023634	012746	000021					
5959	023640	016601	000024					
5960	023644	016600	000022					
5961	023650	100005						
5962	023652	105366	000003					
5963	023656	005400						
5964	023660	005401						
5965	023662	005600						
5966	023664	016602	000020	15:				
5967	023670	002407						
5968	023672	003011						
5969	023674	052766	000003	000014				
5970	023702	012700	177777					
5971	023706	000424						
5972	023710	005266	000002	25:				
5973	023714	000401						
5974	023716	005402		35:				
5975	023720	000241		45:				
5976	023722	000405						
5977	023724	006100		55:				
5978	023726	010003						
5979	023730	060203						
5980	023732	103001						
5981	023734	010300						
5982	023736	006101		65:				
5983	023740	005316						
5984	023742	001370						
5985	023744	005701						
5986	023746	100005						
5987	023750	052766	000002	000014				
5988	023756	005000						
5989	023760	010001		75:				
5990	023762	005726		85:				
5991	023764	005716						
5992	023766	002004						
5993	023770	005400						
5994	023772	105066	000001					
5995	023776	005316						
5996	024000	005726		95:				
5997	024002	001401						
5998	024004	005401						
5999	024006	010166	000020	105:				
6000	024012	010066	000016					
6001	024016	012603						
6002	024020	012602						
6003	024022	012601						



```

6004 024024 012600          MOV      (SP)+, R0          ;; POP STACK INTO R0
6005 024026 012666 000002  MOV      (SP)+, 2(SP)      ;; SETUP TO RETURN CONDITION CODES
6006 024032 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 024034 005037 001230  LP. AVL: CLR      @#LPTAVL          ; START WITH NO PRINTER AVAIABLE
6018 024040 012737 024064 000004  MOV      #15, @#ERRVEC      ; SETUP THE TIMEOUT VECTOR
6019 024046 005037 000006          CLR      @#ERRVEC+2
6020 024052 005777 155342          TST      @LPS              ; IS THERE A LINE PRINTER?
6021 024056 005237 001230          INC      @#LPTAVL          ; YES--SET AVAILABLE SWITCH
6022 024062 000401          BR      2$
6023 024064 022626          1$:  CMP      (SP)+, (SP)+      ; NO--POP STACK
6024 024066 012737 000006 000004  2$:  MOV      #ERRVEC+2, @#ERRVEC ; RESTORE TIMEOUT VECTOR
6025 024074 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 024076 010146          ST. CLK: MOV      R1, -(SP)          ; SAVE R1
6044 024100 012701 000006  MOV      #ERRVEC+2, R1      ; SAVE AND SETUP TIMEOUT VECTOR
6045 024104 011146          MOV      (R1), -(SP)
6046 024106 005011          CLR      (R1)              ; LEVEL 0
6047 024110 014146          MOV      -(R1), -(SP)
6048 024112 012711 024142  MOV      #15, (R1)          ; GO TO 1$ ON TIMEOUT
6049 024116 005037 001244  CLR      CLKSTA            ; SET CLOCK STATUS TO NO CLOCK
6050 024122 005777 155252          TST      @PKCS            ; IS THERE A KW11-P?
6051 024126 012737 000001 001244  MOV      #1, CLKSTA        ; YES--SET STATUS TO KW11-P
6052 024134 004737 024244          JSR      PC, ST. PCLK      ; START THE KW11-P
6053 024140 000414          BR      3$                ; GO TO EXIT
6054 024142 022626          1$:  CMP      (SP)+, (SP)+      ; CLEAN UP THE STACK
6055 024144 012711 024170  MOV      #25, (R1)          ; IF TIMEOUT GO TO 2$
6056 024150 005777 155236          TST      @LKS              ; IS THERE A KW11-L?
6057 024154 012737 177777 001244  MOV      #-1, CLKSTA        ; YES-- SET STATUS TO KW11-L
6058 024162 004737 024306          JSR      PC, ST. LCLK      ; START THE KW11-L
6059 024166 000401          BR      3$                ; EXIT
    
```

```

6060 024170 022626      25:  CMP      (SP)+, (SP)+      ; CLEAN UP THE STACK
6061 024172 012621      35:  MOV      (SP)+, (R1)+      ; RESTORE THE TIMEOUT VECTOR
6062 024174 012621      MOV      (SP)+, (R1)+
6063 024176 012601      MOV      (SP)+, R1        ; RESTORE R1
6064 024200 032737 000100 001220  BIT      #SW06, @C. SWR    ; 50HZ OR 60HZ?
6065 024206 001407      BEQ      45                ; BRANCH IF 60
6066 024210 012737 000020 001246  MOV      #20, @TICKMS     ; SETUP TIME PER
6067 024216 012737 047040 001250  MOV      #20000, @TICKUS  ; TICK FOR 50HZ
6068 024224 000406      BR       55
6069 024226 012737 000016 001246  45:  MOV      #16, @TICKMS     ; SETUP TIME PER
6070 024234 012737 040432 001250  MOV      #16666, @TICKUS  ; TICK FOR 60HZ
6071 024242 000207      55:  RTS      PC                ; RETURN
6072
6073 024244      ST. PCLK:
6074 024244 032737 000040 001220  BIT      #SW05, @C. SWR    ; ALLOW SOFTWARE TIMEOUTS?
6075 024252 001014      BNE      15                ; NO--BRANCH
6076 024254 012777 024342 155112  MOV      #SRVCLK, @PKV    ; SETUP THE KW11-P VECTOR
6077 024262 012777 000300 155106  MOV      #300, @PKV+2
6078 024270 012777 000001 155104  MOV      #1, @PKB         ; COUNT ONE TICK
6079 024276 012777 000115 155074  MOV      #115, @PKCS     ; "INT. EN.", "COUNT DOWN", "MODE 1 (REPEAT)",
6080                                     ; "LINE FREQ", AND "RUN"
6081 024304 000207      15:  RTS      PC                ; RETURN
6082
6083 024306      ST. LCLK:
6084 024306 032737 000040 001220  BIT      #SW05, @C. SWR    ; ALLOW SOFTWARE TIMEOUTS?
6085 024314 001011      BNE      15                ; NO--BRANCH
6086 024316 012777 024342 155062  MOV      #SRVCLK, @LKV    ; SETUP THE KW11-L VECTOR
6087 024324 012777 000300 155056  MOV      #300, @LKV+2
6088 024332 012777 000100 155052  MOV      #100, @LKS      ; START THE KW11-L
6089 024340 000207      15:  RTS      PC                ; RETURN
6090
6091 024342 013746 001246  SRVCLK: MOV      @TICKMS, -(SP) ; TIME PER TICK IN MILLISECONDS
6092 024346 004737 040736  JSR      PC, @RPTMR       ; COUNT THE ELAPSED TIME
6093 024352 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 LODFLT:
6102 024354 010046  MOV      R0, -(SP)        ; PUSH R0 ON STACK
6103 024356 010146  MOV      R1, -(SP)        ; PUSH R1 ON STACK
6104 024360 010246  MOV      R2, -(SP)        ; PUSH R2 ON STACK
6105 024362 010346  MOV      R3, -(SP)        ; PUSH R3 ON STACK
6106 024364 012737 176777 001234  MOV      #176777, TSTNMS  ; SELECT TESTS 0-10, 12-17
6107 024372 012737 000001 001236  MOV      #1, TSTNMS+2    ; SET SELECT BIT FOR TEST 20
6108 024400 012700 001664  MOV      #DFLT, R0        ; DEFAULT PARAMETERS POINTER
6109 024404 012701 002330  MOV      #PRMO, R1       ; TABLE POINTER
6110 024410 010102  MOV      R1, R2          ; STOP ADDRESS
6111 024412 012021  15:  MOV      (R0)+, (R1)+    ; MOVE DEFAULT PARAMETERS INTO
6112 024414 020002  CMP      R0, R2          ; RUN TIME TABLES ** DONE?
6113 024416 103775  BLO      15                ; NO--BRANCH
6114 024420 012700 003504  MOV      #PAT8, R0       ; PAT0 DEFAULTS TO PATTERN 8
6115 024424 012701 003104  MOV      #PATO, R1

```



```

6116 024430 012021          25:  MOV      (R0)+, (R1)+
6117 024432 020027 003544    CMP      RO, #PAT9
6118 024436 103774          BLO     25
6119 024440 032737 000001 001220  BIT     #BIT00, C.SWR ;16 BIT MODE ?
6120 024446 001012          BNE     35 ;BR IF 18
6121 024450 012737 000025 001630  MOV     #21, PRMLMT+22 ;SET 'FS' LIMIT TO 21.
6122 024456 012737 000025 001632  MOV     #21, PRMLMT+24 ;SET 'LS' LIMIT TO 21.
6123 024464 012737 165000 001352  MOV     #-<256, #22, >, TRCKWC ;WORD COUNT FOR A 16 BIT TRACK
6124 024472 000411          BR      45 ;CONTINUE
6125 024474 012737 000023 001630 35:  MOV     #19, PRMLMT+22 ;SET 'FS' LIMIT TO 19.
6126 024502 012737 000023 001632  MOV     #19, PRMLMT+24 ;SET 'LS' LIMIT TO 19.
6127 024510 012737 166000 001352  MOV     #-<256, #20, >, TRCKWC ;WORD COUNT FOR COUNT FOR AN 18 BIT TRACK
6128 024516 012701 001536          45:  MOV     #PRMPT, R1 ;ADDRESS OF PARAMETER POINTER TABLE
6129 024522 005711          55:  TST     (R1) ;END OF THE TABLE ?
6130 024524 001425          BEQ     85 ;BR IF END
6131 024526 032731 002000  BIT     #BIT10, 2(R1)+ ;'LS' SELECTED ?
6132 024532 001773          BEQ     55 ;BR IF NOT
6133 024534 016102 177776          MOV     -2(R1), R2 ;PARAMETER TABLE ADDRESS
6134 024540 011246          MOV     (R2), -(SP) ;PARAMETER ALLOCATION BITS
6135 024542 012703 000013  MOV     #11, R3 ;NUMBER OF PARAMETERS (MAXIMUM) BEFORE 'LS'
6136 024546 006216          65:  ASR     (SP) ;COUNT THE PARAMETER
6137 024550 103002          BCC     75 ;BR IF NOT USED
6138 024552 062702 000002  ADD     #2, R2 ;INCREMENT THE PARAMETER TABLE ADDRESS
6139 024556 005303          75:  DEC     R3 ;COUNT THE PARAMETER
6140 024560 001372          BNE     65 ;BR IF NOT THERE YET
6141 024562 005726          TST     (SP)+ ;CORRECT THE STACK POINTER
6142 024564 021237 001630  CMP     (R2), PRMLMT+22 ;IS 'LS' TOO LARGE FOR THE MODE SELECTED ?
6143 024570 101754          BLOS   55 ;BR IF NOT
6144 024572 013712 001630  MOV     PRMLMT+22, (R2) ;RESET VALUE FOR MODE USED
6145 024576 000751          BR      55 ;CONTINUE
6146 024600          85:
6147 024600 012603          MOV     (SP)+, R3 ;;POP STACK INTO R3
6148 024602 012602          MOV     (SP)+, R2 ;;POP STACK INTO R2
6149 024604 012601          MOV     (SP)+, R1 ;;POP STACK INTO R1
6150 024606 012600          MOV     (SP)+, R0 ;;POP STACK INTO R0
6151 024610 000207          RTS     PC ;RETURN
6152
6153 ; THIS ROUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST.
6154 ; CALL
6155 ; MOV     #TESTNUM, STSTNM ;LOAD THE TEST NUMBER
6156 ; JSR     PC, LODPRM
6157 ; RETURN
6158
6159 LODPRM:
6160 024612 010146          MOV     R1, -(SP) ;;PUSH R1 ON STACK
6161 024614 010246          MOV     R2, -(SP) ;;PUSH R2 ON STACK
6162 024616 010346          MOV     R3, -(SP) ;;PUSH R3 ON STACK
6163 024620 010446          MOV     R4, -(SP) ;;PUSH R4 ON STACK
6164 024622 005004          CLR     R4 ;CLEAR R4
6165 024624 113704 001102  MOVB   $STSTNM, R4 ;GET THE TEST NUMBER
6166 024630 006304          ASL     R4 ;SETUP TO ADDRESS WORDS
6167 024632 016401 001536  MOV     PRMPT(R4), R1 ;GET THE TEST'S PARAMETER TABLE ADDRESS
6168 024636 012702 001504  MOV     #PRM, R2 ;PARAMETER EXECUTION TABLE
6169 024642 005003          CLR     R3 ;R3 IS USED AS A COUNTER
6170 024644 013704 001254  MOV     CHKDRV, R4 ;DRIVE'S ADDRESS
6171 024650 012122          MOV     (R1)+, (R2)+ ;PARAMETER SPECIFIER
    
```

```

6172 024652 006237 001504      15:   ASR   PRM           ; THIS PARAMETER USED IN THE TEST ?
6173 024656 103002              BCC   25           ; BR IF NOT
6174 024660 012122              MOV   (R1)+, (R2)+ ; LOAD THE VALUE
6175 024662 000401              BR    35           ; CONTINUE
6176 024664 005022      25:   CLR   (R2)+       ; CLEAR THE UNUSED PARAMETER LOCATION
6177 024666 005203      35:   INC   R3          ; COUNT THE POSITION IN THE OUTPUT TABLE
6178 024670 020327 000014      CMP   R3, #12.    ; FINISHED ?
6179 024674 001430              BEQ   65           ; BR IF YES
6180 024676 020327 000003      CMP   R3, #3     ; DOING THE CYLINDER ADDRESSES ?
6181 024702 001363              BNE   15           ; BR IF NOT
6182 024704 132764 000004 034400 BITB  #BIT02, DRVTYP(R4) ; RPO6 ?
6183 024712 001016              BNE   55           ; BR IF IT IS
6184 024714 062703 000002      ADD   #2, R3     ; COUNT THE BYPASSED PARAMETERS (FC' & LC')
6185 024720 006237 001504      ASR   PRM        ; SHIFT THE COUNTER
6186 024724 103002              BCC   45         ; BR IF FC' IS NOT USED
6187 024726 062701 000002      ADD   #2, R1     ; MOVE THE INPUT POINTER
6188 024732 006237 001504      45:   ASR   PRM        ; COUNT THE PARAMETER
6189 024736 103345              BCC   15         ; BR IF LC' NOT USED
6190 024740 062701 000002      ADD   #2, R1     ; MOVE THE INPUT PINTER
6191 024744 000742              BR    15         ; KEEP GOING
6192 024746 000741              BR    15         ; KEEP GOING
6193 024750 162702 000004      55:   SUB   #4, R2    ; BACKUP THE OUTPUT POINTER
6194 024754 000736              BR    15         ; KEEP GOING
6195 024756              65:
6196 024756 012604              MOV   (SP)+, R4   ; POP STACK INTO R4
6197 024760 012603              MOV   (SP)+, R3   ; POP STACK INTO R3
6198 024762 012602              MOV   (SP)+, R2   ; POP STACK INTO R2
6199 024764 012601              MOV   (SP)+, R1   ; POP STACK INTO R1
6200 024766 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 024770 032737 000200 001220 LDCMD: BIT   #SW07, @#C. SWR ; DO EXPLICIT SEEKS?
6210 024776 001007              BNE   15         ; YES--BRANCH
6211 025000 012737 000173 004126      MOV   @READHD, @#DPB. B+2 ; NO--SET UP FOR READ HEADER AND
6212 025006 012737 000173 004146      MOV   @READHD, @#DPB. C+2 ; DATA COMMAND
6213 025014 000406              BR    25
6214 025016 012737 000105 004126      15:   MOV   @SEEK, @#DPB. B+2 ; SETUP FOR SEEK COMMAND
6215 025024 012737 000105 004146      MOV   @SEEK, @#DPB. C+2
6216 025032 000207      25:   RTS   PC
6217
6218 ; THIS ROUTINE WILL CALL THE RPO4/5/6 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   RO, @#CALL. A
6223 ; RETURN
6224
6225 025034 005037 001206      CALL. A: CLR   @#ESCAPE ; NO ESCAPE ADDRESS
6226 025040 004037 035304      JSR   RO, @#RPO4 ; CALL RPO4 DRIVER
6227 025044 004104      DPB. A
  
```



```

6228 025046 000772          BR      CALL. A
6229 025050 005737 004122 15:    TST    @DPB. A+16      ; DONE?
6230 025054 001775          BEQ    15              ; NO--LOOP
6231 025056 100032          BPL    35              ; BRANCH IF NO ERROR
6232 025060 012737 025134 001206  MOV    @25, $ESCAPE    ; ESCAPE TO 25 ON ERROR
6233 025066 013737 004116 001270  MOV    @DPB. A+12, @CYL. DS ; CYLINDER
6234 025074 113737 004115 001274  MOV    @DPB. A+11, @TRK. DS ; TRACK
6235 025102 113737 004114 001272  MOV    @DPB. A+10, @SEC. DS ; SECTOR
6236 025110 012746 004122          MOV    @DPB. A+16, -(SP) ; STATUS/ERROR INDICATOR ADDRESS
6237 025114 004737 026250          JSR    PC, @RERINDX    ; FORM DISPATCH INDEX
6238 025120 062607          ADD    (SP)+, PC      ; REPORT PROPER ERROR
6239 025122 104041          ERROR 41              ;
6240 025124 104042          ERROR 42              ; PARITY ERROR
6241 025126 104043          ERROR 43              ; UNSAFE ERROR
6242 025130 104044          ERROR 44              ; NON-I/O ERROR
6243 025132 104045          ERROR 45              ; I/O ERROR
6244 025134 013746 004122 25:    MOV    DPB. A+16, -(SP) ; STATUS WORD
6245 025140 004737 026210          JSR    PC, LOP. CK     ; SEE IF LOOP, ABORT, OR CONTINUE
6246 025144 000200          35:    RTS     RO            ; RETURN
  
```

; THIS ROUTINE IS THE SAME AS "CALL. A" EXCEPT FOR THE DPB USED AND IF  
 ; THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,  
 ; AND SECTOR) READ IS CHECKED FOR VALIDITY.

```

6251 ; CALL
6252 ;
6253 ;       FILL DPB
6254 ;       JSR    RO, @CALL. B
6255 ;       RETURN
  
```

```

6256 025146 005037 001206  CALL. B: CLR    @SESCAPE    ; NO ESCAPE ADDRESS
6257 025152 004037 035304          JSR    RO, @RPO4      ; CALL RPO4 DRIVER
6258 025156 004124          DPB. B
6259 025160 000772          BR      CALL. B
6260 025162 005737 004142 15:    TST    DPB. B+16      ; DONE?
6261 025166 001775          BEQ    15              ; NO--BRANCH
6262 025170 100042          BPL    45              ; BRANCH IF NO ERROR
6263 025172 012737 025264 001206  MOV    @35, $ESCAPE    ; ESCAPE TO 35 ON ERROR
6264 025200 013737 004136 001270  MOV    @DPB. B+12, @CYL. DS ; CYLINDER
6265 025206 113737 004135 001274  MOV    @DPB. B+11, @TRK. DS ; TRACK
6266 025214 113737 004134 001272  MOV    @DPB. B+10, @SEC. DS ; SECTOR
6267 025222 012746 004142          MOV    @DPB. B+16, -(SP) ; STATUS/ERROR INDICATOR ADDRESS
6268 025226 004737 026250          JSR    PC, @RERINDX    ; FORM DISPATCH INDEX
6269 025232 062607          ADD    (SP)+, PC      ; REPORT PROPER ERROR
6270 025234 104041          ERROR 41              ;
6271 025236 104042          ERROR 42              ; PARITY ERROR
6272 025240 104043          ERROR 43              ; UNSAFE ERROR
6273 025242 104044          ERROR 44              ; NON-I/O ERROR
6274 025244 005737 004220          TST    RP. REG+RPER1   ; DRIVE ERROR ?
6275 025250 001404          BEQ    25              ; BR IF NOT
6276 025252 032737 177677 004220  BIT    # C100, RP. REG+RPER1 ; SEE IF ONLY 'HCE' SET
6277 025260 001406          BEQ    45              ; BR IF IT IS
6278 025262 104045          25:    ERROR 45              ; I/O ERROR
6279 025264 013746 004142 35:    MOV    DPB. B+16, -(SP) ; STATUS WORD
6280 025270 004737 026210          JSR    PC, LOP. CK     ; SEE IF LOOP, ABORT, OR CONTINUE
6281 025274 000410          BR      55              ; CHECK FOR STALL
6282 025276 123727 004126 000173 45:    MOV    @DPB. B+2, @READHD ; DOING IMPLIED SEEKS?
6283 025304 001004          BEQ    55              ; NO--BRANCH
  
```



```

6284 025306 004037 026530      JSR    RO, @VERIFY      ; YES--GO CHECK THE DATA
6285 025312 004134              DPB. B+10
6286 025314 000407              BR     65               ; ERROR DURING VERIFY
6287 025316 032737 040000 001220 55:  BIT    @SW14, @C. SWR   ; STALL?
6288 025324 001403              BEQ    65               ; NO--BRANCH
6289 025326 004037 026366      JSR    RO, @STALL      ; YES--CALL STALL ROUTINE
6290 025332 001354              .WORD STALL1          ; STALL TIME POINTER
6291 025334 000200 65:    RTS     RO         ; RETURN
6292
6293 ; THIS ROUTINE IS THE SAME AS "CALL. B" EXCEPT FOR THE DPB USED.
6294 ; CALL
6295 ;
6296 ;     FILL DPB
6297 ;     JSR    RO, @CALL. C
6298 ;     RETURN
6299 025336 005037 001206      CALL. C: CLR    @#SESCAPE ; NO ESCAPE ADDRESS
6300 025342 004037 035304      JSR    RO, @RPO4      ; CALL RPO4 DRIVER
6301 025346 004144              DPB. C
6302 025350 000772              BR     CALL. C
6303 025352 005737 004162 15:    TST    @DPB. C+16    ; DONE?
6304 025356 001775              BEQ    15              ; NO--LOOP
6305 025360 100042              BPL    45              ; YES--BRANCH IF NO ERROR
6306 025362 012737 025454 001206  MOV    @35, @SESCAPE ; ESCAPE TO 35 ON ERROR
6307 025370 013737 004156 001270  MOV    @DPB. C+12, @CYL. DS ; CYLINDER
6308 025376 113737 004155 001274  MOVB   @DPB. C+11, @TRK. DS ; TRACK
6309 025404 113737 004154 001272  MOVB   @DPB. C+10, @SEC. DS ; SECTOR
6310 025412 012746 004162      MOV    @DPB. C+16, -(SP) ; STATUS/ERROR INDICATOR ADDRESS
6311 025416 004737 026250      JSR    PC, @ERRINDX   ; FORM DISPATCH INDEX
6312 025422 062607      ADD    (SP)+, PC      ; REPORT PROPER ERROR
6313 025424 104041      ERROR  41
6314 025426 104042      ERROR  42             ; PARITY ERROR
6315 025430 104043      ERROR  43             ; UNSAFE ERROR
6316 025432 104044      ERROR  44             ; NON-I/O ERROR
6317 025434 005737 004220      TST    RP. REG+RPER1 ; DRIVE ERROR ?
6318 025440 001404      BEQ    25             ; BR IF NOT
6319 025442 032737 177677 004220  BIT    # C100, RP. REG+RPER1 ; SEE IF ONLY 'HCE' SET
6320 025450 001406      BEQ    45             ; BR IF IT IS
6321 025452 104045 25:    ERROR  45             ; I/O ERROR
6322 025454 013746 004162 35:    MOV    DPB. C+16, -(SP) ; STATUS WORD
6323 025460 004737 026210      JSR    PC, LOP. CK    ; SEE IF LOOP, ABORT, OR CONTINUE
6324 025464 000410      BR     55
6325 025466 123727 004146 000173 45:    CMPB   @DPB. C+2, @READHD ; DOING IMPLIED SEEK?
6326 025474 001004      BNE    55             ; NO--EXIT
6327 025476 004037 026530      JSR    RO, @VERIFY    ; YES--CHECK THE DATA
6328 025502 004154              DPB. C+10
6329 025504 000407              BR     65               ; ERROR DURING VERIFY
6330 025506 032737 040000 001220 55:  BIT    @SW14, @C. SWR   ; STALL?
6331 025514 001403              BEQ    65               ; NO--BRANCH
6332 025516 004037 026366      JSR    RO, @STALL      ; YES--CALL STALL ROUTINE
6333 025522 001354              .WORD STALL1          ; STALL TIME POINTER
6334 025524 000200 65:    RTS     RO         ; RETURN
6335
6336 ; THIS ROUTINE IS THE SAME AS "CALL. A" EXCEPT FOR THE DPB USED AND
6337 ; ON AN ERROR LOCATION "ERR. CT" IS EXAMINED. IF ERR. CT IS EQUAL TO
6338 ; SERFLG EXIT IS TO THE NEXT TEST.
6339

```



```

6340 ;CALL
6341 ; FILL DPB
6342 ; JSR RO,@#DRVCAL
6343 ; RETURN
6344
6345 025526 005037 001206 DRVCAL: CLR @#SESCAPE ;NO ESCAPE ADDRESS
6346 025532 005037 001334 CLR @#WCEFLG ;CLEAR WRITE CHECK ERROR FLAG
6347 025536 004037 035304 JSR RO,@#RPO4 ;CALL RPO4 DRIVER
6348 025542 004164 DTADPB
6349 025544 000770 BR DRVCAL
6350 025546 005737 004202 DRVCL1: TST @#DTADPB+16 ;DONE
6351 025552 001775 BEQ DRVCL1 ;NO--LOOP
6352 025554 100402 BMI 15 ;BR IF ERRORS
6353 025556 000137 026170 JMP 105 ;NO ERRORS
6354 025562
6355 025562 012737 025636 001206 15: MOV @25,SESCAPE ;;ESCAPE TO 25 ON ERROR
6356 025570 013737 004176 001270 MOV @#DTADPB+12,@#CYL.DS ;CYLINDER
6357 025576 113737 004175 001274 MOV @#DTADPB+11,@#TRK.DS ;TRACK
6358 025604 113737 004174 001272 MOV @#DTADPB+10,@#SEC.DS ;SECTOR
6359 025612 012746 004202 MOV @#DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6360 025616 004737 026250 JSR PC,@#ERINDX ;FORM DISPATCH INDEX
6361 025622 062607 ADD (SP)+,PC ;REPORT PROPER ERROR
6362 025624 104041 ERROR 41 ;
6363 025626 104042 ERROR 42 ;PARITY ERROR
6364 025630 104043 ERROR 43 ;UNSAFE ERROR
6365 025632 104044 ERROR 44 ;NON-1/0 ERROR
6366 025634 104045 ERROR 45 ;1/0 ERROR
6367 025636 122737 000020 001102 25: CMPB @20,@#STSTNM ;TEST 20?
6368 025644 001137 BNE 85 ;NO--BRANCH
6369 025646 013746 004202 MOV DTADPB+16,-(SP) ;STATUS WORD
6370 025652 004737 026210 JSR PC,LOP.CK ;SEE IF LOOP, ABORT, OR CONTINUE
6371 025656 122737 000151 004166 CMPB @#WRCKD,@#DTADPB+2 ;DOING A WRITE CHECK?
6372 025664 001133 BNE 125 ;NO--BRANCH
6373 025666 032737 040000 004214 BIT @BIT14,@#RP.REG+10 ;IS "WCE"=1?
6374 025674 001527 BEQ 125 ;NO--BRANCH
6375 025676 032777 000020 153234 BIT @#SW04,@#SHR ;INHIBIT WRITES?
6376 025704 001123 BNE 125 ;YES--BRANCH
6377 025706 112737 000161 004166 MOV @#WRITE,@#DTADPB+2 ;SETUP FOR A WRITE
6378 025714 005037 001206 CLR @#SESCAPE ;NO ESCAPE ADDRESS
6379 025720 004037 035304 JSR RO,@#RPO4 ;DO THE WRITE
6380 025724 004164 DTADPB
6381 025726 000240 NOP
6382 025730 005737 004202 35: TST @#DTADPB+16 ;DONE?
6383 025734 001775 BEQ 35 ;NO--LOOP
6384 025736 100026 BPL 45 ;YES--BRANCH IF NO ERROR
6385 025740 012737 026144 001206 MOV @85,SESCAPE ;;ESCAPE TO 85 ON ERROR
6386 025746 013737 004176 001270 MOV @#DTADPB+12,@#CYL.DS ;CYLINDER
6387 025754 113737 004175 001274 MOV @#DTADPB+11,@#TRK.DS ;TRACK
6388 025762 113737 004174 001272 MOV @#DTADPB+10,@#SEC.DS ;SECTOR
6389 025770 012746 004202 MOV @#DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
6390 025774 004737 026250 JSR PC,@#ERINDX ;FORM DISPATCH INDEX
6391 026000 062607 ADD (SP)+,PC ;REPORT PROPER ERROR
6392 026002 104041 ERROR 41 ;
6393 026004 104042 ERROR 42 ;PARITY ERROR
6394 026006 104043 ERROR 43 ;UNSAFE ERROR
6395 026010 104044 ERROR 44 ;NON-1/0 ERROR
  
```

```

6396 026012 104045          ERROR 45          ; I/O ERROR
6397 026014 112737 000151 004166 45:  MOVB  #WRCKD, @DTADPB+2 ; COMMAND=WRITE CHECK DATA
6398 026022 004037 035304          JSR   RO, @RPO4      ; DO THE WRITE CHECK
6399 026026 004164          DTADPB
6400 026030 000240          NOP
6401 026032 005737 004202          55:  TST   @DTADPB+16     ; DONE?
6402 026036 001775          BEQ   55             ; NO--LOOP
6403 026040 100410          BMI   75             ; YES--BRANCH IF ERROR
6404 026042 004037 035304          JSR   RO, @RPO4      ; DO A 2ND WRITE CHECK
6405 026046 004164          DTADPB
6406 026050 000240          NOP
6407 026052 005737 004202          65:  TST   @DTADPB+16     ; DONE?
6408 026056 001775          BEQ   65             ; NO--LOOP
6409 026060 100043          BPL   105            ; YES--BRANCH IF NO ERROR
6410 026062 012737 000001 001334 75:  MOV   #1, @WCEFLG     ; SET THE WRITE CHECK ERROR FLAG
6411 026070 012737 026144 001206          MOV   #85, $ESCAPE   ; ESCAPE TO 85 ON ERROR
6412 026076 013737 004176 001270          MOV   @DTADPB+12, @CYL.DS ; CYLINDER
6413 026104 113737 004175 001274          MOVB  @DTADPB+11, @TRK.DS ; TRACK
6414 026112 113737 004174 001272          MOVB  @DTADPB+10, @SEC.DS ; SECTOR
6415 026120 012746 004202          MOV   @DTADPB+16, -(SP) ; STATUS/ERROR INDICATOR ADDRESS
6416 026124 004737 026250          JSR   PC, @ERINDX    ; FORM DISPATCH INDEX
6417 026130 062607          ADD   (SP)+, PC      ; REPORT PROPER ERROR
6418 026132 104041          ERROR 41            ;
6419 026134 104042          ERROR 42            ; PARITY ERROR
6420 026136 104043          ERROR 43            ; UNSAFE ERROR
6421 026140 104044          ERROR 44            ; NON-I/O ERROR
6422 026142 104046          ERROR 46            ; FATAL WRITE CHECK
6423 026144 013746 004202          85:  MOV   DTADPB+16, -(SP) ; STATUS WORD
6424 026150 004737 026210          JSR   PC, LOP.CK     ; SEE IF LOOP, ABORT, OR CONTINUE
6425 026154 123737 001364 001103 125:  CMPB  @ERR.CT, @SERFLG ; GO TO NEXT TEST?
6426 026162 101002          BHI   105            ; NO--BRANCH
6427 026164 013700 001252          95:  MOV   @BYPASS, RO    ; YES--GET EXIT ADDRESS
6428 026170 032737 040000 001220 105:  BIT   #SW14, @C.SWR  ; STALL?
6429 026176 001403          BEQ   115            ; NO--BRANCH
6430 026200 004037 026366          JSR   RO, @STALL     ; YES--CALL STALL ROUTINE
6431 026204 001356          .WORD STALL2        ; STALL TIME POINTER
6432 026206 000200          115: RTS   RO
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 026210 032777 001000 152722 LOP.CK: BIT   #SW9, @SWR      ; LOOP ON ERROR
6442 026216 001402          BEQ   15             ; BR IF NOT
6443 026220 000177 152664          JMP   @SLPERR        ; START AT THE LOOP ADDRESS
6444 026224 005037 001206          15:  CLR   $ESCAPE        ; CLEAR ERROR ESCAPE FLAG
6445 026230 032766 072006 000002          BIT   #BIT14!BIT13!BIT12!BIT10!BIT02!BIT01, 2(SP) ; CHECK ERROR TYPE
6446 026236 001402          BEQ   25             ; BR IF DRIVE NOT OFFLINE, UNLOADED, OR
6447 ; PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
6448 026240 000137 017366          JMP   $EOP           ; TERMINATE DRIVE
6449 026244 012616          25:  MOV   (SP)+, (SP)    ; ADJUST RETURN ADDRESS
6450 026246 000207          RTS   PC
6451
    
```



```

6452 ; THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
6453 ; TO THE PROPER ERROR CALL. THE INDEX IS FORMED BY EXAMINING
6454 ; THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.
6455 ; INDEX STATUS/ERROR
6456 ;-----
6457 ; 0 BIT14!BIT13!BIT08!BIT01
6458 ; 2 BIT11!BIT10!BIT02
6459 ; 4 BIT12!BIT04
6460 ; 6 BIT05!BIT03!<BIT09 & COMMAND=NON-I/O>
6461 ; 10 BIT06!<BIT09 & COMMAND=I/O>
6462 ; CALL
6463 ; JSR #DPB+16,-(SP) ; ADDRESS OF STATUS/ERROR INDICATOR
6464 ; JSR PC,@#ERINDX ; FORM INDEX
6465 ; RETURN ; INDEX IS ON THE STACK
6466
6467 026250 010046 ERINDX: MOV R0,-(SP) ; SAVE R0
6468 026252 010146 MOV R1,-(SP) ; SAVE R1
6469 026254 016600 000006 MOV 6(SP),R0 ; GET STATUS/ERROR INDICATOR POINTER
6470 026260 011037 001260 MOV (R0),@#SVSTAT ; SAVE THE STATUS/ERROR INDICATOR
6471 026264 005001 CLR R1 ; START INDEX AT ZERO
6472 026266 032710 BIT (PC)+,(R0) ; FORM INDEX OF 0?
6473 026270 020402 .WORD BIT13!BIT08!BIT01
6474 026272 001027 BNE 55 ; YES--BRANCH
6475 026274 032710 BIT (PC)+,(R0) ; FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
6476 026276 006004 .WORD BIT11!BIT10!BIT02
6477 026300 001023 BNE 45 ; YES--BRANCH
6478 026302 032710 BIT (PC)+,(R0) ; FORM UNSAFE INDEX (4)?
6479 026304 050020 .WORD BIT14!BIT12!BIT04
6480 026306 001017 BNE 35 ; YES--BRANCH
6481 026310 032710 BIT (PC)+,(R0) ; FORM NON-I/O ERROR INDEX (6)?
6482 026312 000050 .WORD BIT05!BIT03
6483 026314 001013 BNE 25 ; YES--BRANCH
6484 026316 032710 BIT (PC)+,(R0) ; FORM I/O ERROR INDEX (10)?
6485 026320 000100 .WORD BIT06
6486 026322 001007 BNE 15 ; YES--BRANCH
6487 026324 032710 BIT (PC)+,(R0) ; SOFTWARE TIMEOUT?
6488 026326 001000 .WORD BIT09
6489 026330 001410 BEQ 55 ; NO--FORM INDEX OF 0
6490 026332 122760 000150 177762 CMPB #150,-16(R0) ; YES--I/O?
6491 026340 003001 BGT 25 ; NO--BRANCH
6492 026342 005201 15: INC R1 ; INDEX=10---ERROR=45 OR 46
6493 026344 005201 25: INC R1 ; INDEX=6---ERROR=44
6494 026346 005201 35: INC R1 ; INDEX=4---ERROR=43
6495 026350 005201 45: INC R1 ; INDEX=2---ERROR=42
6496 026352 006301 55: ASL R1 ; INDEX=0---ERROR=41
6497 026354 010166 000006 MOV R1,6(SP) ; RETURN INDEX TO USER
6498 026360 012601 MOV (SP)+,R1 ; RESTORE R1
6499 026362 012600 MOV (SP)+,R0 ; RESTORE R0
6500 026364 000207 RTS PC ; RETURN FROM CALL

```

```

6501 ; THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
6502 ; AMOUNT OF TIME IF BIT13 OF C. SWR = 0 OR A RANDOM AMOUNT OF TIME
6503 ; IF BIT 13 OF C. SWR = 1.
6504 ; STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0 - 7, AND STALL2
6505 ; CONTAINS THE TIME FOR TESTS 16-21.
6506 ; CALL
6507

```

```

6508 ; JSR RO,@#STALL
6509 ; TIME POINTER ;WHERE TO FIND THE STALL TIME
6510 ;
6511 STALL: MOV @ (RO)+, -(SP) ; PICKUP STALL TIME
6512 026370 032737 020000 001220 BIT #SW13, @#C. SWR ; USE A RANDOM TIME?
6513 026376 001406 BEQ 1$ ; NO--BRANCH
6514 026400 004737 023512 JSR PC, @#SRAND ; YES--FORM RANDOM NUMBER
6515 026404 013716 023612 MOV @#$LONUM, (SP) ; AND USE IT FOR THE STALL TIME
6516 026410 042716 177700 BIC # (77, (SP) ; BUT NEVER > 64 MILLISECONDS
6517 026414 005046 1$: CLR -(SP) ; CLEAR TEMP. LOCATION
6518 026416 162766 000001 000002 2$: SUB #1, 2(SP) ; MORE STALL REQUIRED?
6519 026424 103407 BLO 4$ ; NO--BRANCH
6520 026426 012716 000144 MOV #100, (SP) ; STALL FOR ABOUT 1 MILLISECOND
6521 026432 005700 3$: TST RO ; NOP TO KILL TIME
6522 026434 005366 000000 DEC 0(SP) ; COUNT
6523 026440 001374 BNE 3$ ; LOOP IF MORE COUNTS NEEDED
6524 026442 000765 BR 2$
6525 026444 022626 4$: CMP (SP)+, (SP)+ ; CLEAN OFF THE STACK
6526 026446 000200 RTS RO ; EXIT
6527
6528
6529 ; ROUTINE TO PROVIDE A 2 MS STALL AFTER A SEEK OPERATION IN THE SEEK TIMING
6530 ; TESTS. THIS STALL IS REQUIRED TO COMPENSATE FOR THE 'ACCESS READY' DELAY
6531 ; IN THE RPO4. THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES.
6532 ; CALL
6533 ; JSR PC, @#TWOMS
6534 ; RETURN
6535
6536 THOMS: MOV @#PS, -(SP) ; SAVE THE PRESENT PROCESSOR STATUS
6537 026450 013746 177776 MOV #(<5*32.), @#PS ; SET THE PROCESSOR PRIORITY TO 5
6538 026454 012737 000240 177776 MOV @#PKV, -(SP) ; SAVE THE OLD CLOCK VECTOR ADDRESS
6539 026462 017746 152706 MOV #1$, @#PKV ; SETUP NEW VECTOR ADDRESS
6540 026466 012777 026512 152700 MOV #200, @#PKB ; LOAD THE CLOCK BUFFER
6541 026474 012777 000310 152700 MOV #101, @#PKCS ; START THE CLOCK
6542 026502 012777 000101 152670 WAIT ; WAIT FOR 2 MS
6543 026510 000001 1$: ADD #4, SP ; INCREMENT STACK FOR RETURN
6544 026512 062706 000004 MOV (SP)+, @#PKV ; RESTORE OLD CLOCK VECTOR
6545 026516 012677 152652 MOV (SP)+, @#PS ; RESTORE THE OLD PROCESSOR STATUS
6546 026522 012637 177776 RTS PC ; RETURN
6547
6548 ; ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS
6549 ; CALL
6550 ; JSR RO, @#VERIFY
6551 ; ADR POINTER ; ADDRESS OF DPB+10 (SECTOR NUMBER)
6552 ; RETURN
6553
6554 VERIFY: MOV R1, -(SP) ; SAVE R1
6555 026530 010146 MOV (RO)+, R1 ; GET ADDRESS OF DPB+10
6556 026532 012001 010000 047714 BIC #FMT22, @#BUFFER ; STRIP FORMAT BIT FROM CYLINDER NUMBER
6557 026534 042737 023761 047714 000002 CMP @#BUFFER, 2(R1) ; CYLINDER NUMBER OK?
6558 026542 023761 047714 000002 BNE 1$ ; NO--BRANCH
6559 026550 001003 CMP @#BUFFER+2, (R1) ; YES--HOW ABOUT TRACK/SECTOR?
6560 026552 023711 047716 BEQ 3$ ; BRANCH IF GOOD
6561 026556 001441 1$: MOV @#BUFFER, @#CYL. RD ; SAVE THE EXPECTED AND THE
6562 026560 013737 047714 001262 MOV @#BUFFER+3, @#TRK. RD ; RECIEVED CYLINDER, TRACK,
6563 026566 113737 047717 001264 MOV @#BUFFER+2, @#SEC. RD ; AND SECTOR
6564 026574 113737 047716 001266

```



```

6564 026602 112137 001272      MOV      (R1)+, @SEC. DS
6565 026606 112137 001274      MOV      (R1)+, @TRK. DS
6566 026612 011137 001270      MOV      (R1), @CYL. DS
6567 026616 012737 026630 001206  MOV      @Z5, $ESCAPE      ;; ESCAPE TO Z5 ON ERROR
6568 026624 005740                TST      -(R0)             ; MAKE IT TEST PC+4
6569 026626 104012                ERROR    12                ; REPORT THE ERROR
6570 026630 012737 000107 004106 25:  MOV      @RECAL, DPB. A+2   ; LOAD RECALIBRATE ORDER CODE
6571 026636 004037 025034                JSR      RO, @CALL. A      ; GO EXECUTE THE COMMAND
6572 026642 005037 001206                CLR      $ESCAPE          ; CLEAR ERROR ESCAPE FLAG
6573 026646 032777 001000 152264  BIT      @SW9, @SWR        ; LOOP ON ERROR ?
6574 026654 001404                BEQ      45                ; BR IF NOT
6575 026656 000177 152226                JMP      @SLPERR          ; RETURN TO ERROR LOOP ADDRESS
6576 026662 062700 000002 35:  ADD      @Z, RO            ; INCREMENT RETURN ADDRESS
6577 026666 012601 45:  MOV      (SP)+, R1        ; RESTORE R1
6578 026670 000200                RTS      RO                ; 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      RO, SRCHOO      ; DO A MASSBUS INIT. AND RECAL
6585                ; RETURN1          ; RETURN HERE IF NO ERROR
6586                ; RETURN2          ; RETURN HERE ON ERROR
6587
6588 026672 005001  SRCHOO: CLR      R1            ; INCASE OF ERROR (TYPTIM)
6589 026674 005037 177776      CLR      @APS
6590 026700 012777 037400 005612  MOV      @ISR, @RVEC      ; SETUP INTERRUPT VECTOR
6591 026706 013704 034516      MOV      @RPNADR, R4     ; PICKUP ADDRESS OF RPCS1
6592 026712 012764 000040 000010  MOV      @BIT05, RPCS2(R4) ; MASSBUS INIT.
6593 026720 005037 004174      CLR      @DTADPB+10     ; TRACK=0; SECTOR=0
6594 026724 005037 004176      CLR      @DTADPB+12     ; CYLINDER = 0
6595 026730 012737 000107 004166  MOV      @RECAL, @DTADPB+2 ; COMMAND = RECALIBRATE
6596 026736 005037 001206      CLR      @SEESCAPE      ; NO ESCAPE ADDRESS
6597 026742 004037 035304      JSR      RO, @RPO4      ; CALL THE DRIVER
6598 026746 004164      DTADPB
6599 026750 000440      BR      45              ; QUEUE IS FULL
6600 026752 005737 004202 15:  TST      DTADPB+16      ; WAIT ON DONE
6601 026756 001775      BEQ      15
6602 026760 100030      BPL      35              ; TAKE NORMAL EXIT IF NO ERROR
6603 026762 012737 027036 001206  MOV      @Z5, $ESCAPE      ;; ESCAPE TO Z5 ON ERROR
6604 026770 013737 004176 001270  MOV      @DTADPB+12, @CYL. DS ; CYLINDER
6605 026776 113737 004175 001274  MOV      @DTADPB+11, @TRK. DS ; TRACK
6606 027004 113737 004174 001272  MOV      @DTADPB+10, @SEC. DS ; SECTOR
6607 027012 012746 004202      MOV      @DTADPB+16, -(SP) ; STATUS/ERROR INDICATOR ADDRESS
6608 027016 004737 026250      JSR      PC, @ERINDX     ; FORM DISPATCH INDEX
6609 027022 062607      ADD      (SP)+, PC       ; REPORT PROPER ERROR
6610 027024 104041      ERROR    41
6611 027026 104042      ERROR    42              ; PARITY ERROR
6612 027030 104043      ERROR    43              ; UNSAFE ERROR
6613 027032 104044      ERROR    44              ; NON-I/O ERROR
6614 027034 104045      ERROR    45              ; I/O ERROR
6615 027036 005720 25:  TST      (R0)+          ; ADJUST FOR ERROR EXIT
6616 027040 000404      BR      45              ; GO TO THE EXIT
6617 027042 005064 000006 35:  CLR      RPOA(R4)        ; TRACK AND SECTOR = 0
6618 027046 005064 000034      CLR      RPCA(R4)        ; CYLINDER = 0
6619 027052 000200 45:  RTS      RO                ; RETURN

```

```

6620
6621 ; THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST
6622
6623 027054 000002 DORTI: RTI ; RETURN FROM INTERRUPT
6624
6625 ; THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE "TIMING ROUTINES"
6626 ; CALL
6627 ; JSR PC, @#STRTMR
6628 ; RETURN
6629
6630 027056 104412 STRTMR: SAVREG ; SAVE R0-R5
6631 027060 012700 001276 MOV #TIM.UP,R0 ; START AT TIM.UP (MINIMUM)
6632 027064 012701 001332 MOV #TIM.PT,R1 ; STOP AT TIM.PT
6633 027070 005020 15: CLR (R0)+ ; CLEAR
6634 027072 020001 CMP R0,R1 ; DONE?
6635 027074 103775 BLO 15 ; NO--BRANCH
6636 027076 012710 047714 MOV #BUFFER,(R0) ; SETUP POINTER
6637 027102 012737 077777 001276 MOV #CBIT15,@TIM.UP ; SET MINIMUM TIME TO MAXIMUM
6638 027110 012737 077777 001314 MOV #CBIT15,@TIM.DN ; POSITIVE NUMBER
6639 027116 104413 RESREG ; RESTORE R0-R5
6640 027120 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 027122 012702 001276 COUNT: MOV #TIM.UP,R2 ; PICKUP THE "UP" POINTER
6653 027126 005705 TST R5 ; USE IT?
6654 027130 001402 BEQ 15 ; YES--BRANCH
6655 027132 012702 001314 MOV #TIM.DN,R2 ; NO--PICKUP "DOWN" POINTER
6656 027136 027722 152242 15: CMP @PKC,(R2)+ ; LESS THAN PREVIOUS LOW?
6657 027142 002003 BGE 25 ; NO--BRANCH
6658 027144 017762 152234 177776 MOV @PKC,-2(R2) ; YES--SAVE IT
6659 027152 027763 152226 000004 25: CMP @PKC,4(R3) ; LESS THAN THE LOW LIMIT?
6660 027160 002001 BGE 35 ; NO--BRANCH
6661 027162 005212 INC (R2) ; YES--COUNT IT
6662 027164 005722 35: TST (R2)+ ; ADVANCE THE POINTER
6663 027166 027722 152212 CMP @PKC,(R2)+ ; GREATER THAN PREVIOUS HIGH?
6664 027172 003403 BLE 45 ; NO--BRANCH
6665 027174 017762 152204 177776 MOV @PKC,-2(R2) ; YES--SAVE IT
6666 027202 027763 152176 000006 45: CMP @PKC,6(R3) ; GREATER THAN THE HIGH LIMIT?
6667 027210 003401 BLE 55 ; NO--BRANCH
6668 027212 005212 INC (R2) ; YES--COUNT IT
6669 027214 005722 55: TST (R2)+ ; ADVANCE THE POINTER
6670 027216 067722 152162 ADD @PKC,(R2)+ ; ADD THIS COUNT TO THE TOTAL
6671 027222 005522 ADC (R2)+
6672 027224 005212 INC (R2) ; COUNT THIS READING
6673 027226 022737 056204 001332 CMP #BUFFER+(4*814),@TIM.PT ; SAVE THIS COUNT?
6674 027234 101406 BLOS 65 ; NO--BRANCH
6675 027236 017777 152142 152066 MOV @PKC,@TIM.PT ; YES--WELL SAVE IT THEN

```



```

6676 027244 062737 000002 001332      ADD      #2, @TIM PT      ;ADVANCE THE POINTER
6677 027252 000207                65:      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      RO, @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 027254 012002                TYPTIM:  MOV      (R0)+, R2      ;PICKUP THE TABLE POINTER
6695 027256 032777 000100 151654      BIT      #SW06, @SWR      ;INHIBIT TIME REPORTS?
6696 027264 001145                BNE      75                ;YES--BRANCH
6697 027266 012237 027306                MOV      (R2)+, 25        ;ADDRESS OF MESSAGE NUMBER 1
6698 027272 012205                MOV      (R2)+, R5        ;ADDRESS OF MESSAGE NUMBER 2
6699 027274 012203                MOV      (R2)+, R3        ;PICKUP THE LOW LIMIT
6700 027276 011202                MOV      (R2), R2         ;AND THE HIGH LIMIT
6701 027300 012704 001276                MOV      @TIM. UP, R4     ;PARAMETER POINTER
6702 027304 104401                15:      TYPE      ;TYPE THE MESSAGE
6703 027306 000000                25:      .WORD    0          ;ASCIZ MESSAGE POINTER GOES HERE
6704 027310 005764 000014                TST      14(R4)          ;DID ANY COUNTS OCCUR?
6705 027314 001527                BEQ      65                ;NO--BRANCH
6706 027316 104401 043702                TYPE      ,MSGMIN        ;"MIN="
6707 027322 012446                MOV      (R4)+, -(SP)     ;PUT (R4)+ ON THE STACK
6708 027324 004737 023222                JSR      PC, @$$SB2D      ;CHANGE (R4)+ TO DECIMAL ASCIZ
6709 027330 004737 023452                JSR      PC, @$$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6710 027334 104401 043727                TYPE      ,MSGOUS        ;"O US"
6711 027340 005724                TST      (R4)+           ;ANY SEEKS BELOW THE LOW LIMIT
6712 027342 001421                BEQ      35                ;NO--BRANCH
6713 027344 104401 044042                TYPE      ,MSG. SP      ;" "
6714 027350 016446 177776                MOV      -2(R4), -(SP)   ;PUT -2(R4) ON THE STACK
6715 027354 004737 023222                JSR      PC, @$$SB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
6716 027360 004737 023452                JSR      PC, @$$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6717 027364 104401 043734                TYPE      ,M BELOW      ;"BELOW THE MINIMUM OF"
6718 027370 010346                MOV      R3, -(SP)       ;PUT R3 ON THE STACK
6719 027372 004737 023222                JSR      PC, @$$SB2D      ;CHANGE R3 TO DECIMAL ASCIZ
6720 027376 004737 023452                JSR      PC, @$$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6721 027402 104401 043727                TYPE      ,MSGOUS        ;"MAX="
6722 027406 104401 043711                35:      TYPE      ,MSGMAX
6723 027412 012446                MOV      (R4)+, -(SP)   ;PUT (R4)+ ON THE STACK
6724 027414 004737 023222                JSR      PC, @$$SB2D      ;CHANGE (R4)+ TO DECIMAL ASCIZ
6725 027420 004737 023452                JSR      PC, @$$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6726 027424 104401 043727                TYPE      ,MSGOUS        ;"BELOW THE MINIMUM OF"
6727 027430 005724                TST      (R4)+           ;ANY SEEKS ABOVE THE HIGH LIMIT
6728 027432 001421                BEQ      45                ;NO--BRANCH
6729 027434 104401 044042                TYPE      ,MSG. SP      ;YES--REPORT HOW MANY
6730 027440 016446 177776                MOV      -2(R4), -(SP)   ;PUT -2(R4) ON THE STACK
6731 027444 004737 023222                JSR      PC, @$$SB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ

```

```

6732 027450 004737 023452      JSR    PC, @#SSUPRS      ;TYPE WITHOUT LEADING ZEROS
6733 027454 104401 043763      TYPE   ,MABOVE          ;"ABOVE THE MAXIMUM OF"
6734 027460 010246              MOV    R2, -(SP)        ;PUT R2 ON THE STACK
6735 027462 004737 023222      JSR    PC, @#SSB2D      ;CHANGE R2 TO DECIMAL ASCIZ
6736 027466 004737 023452      JSR    PC, @#SSUPRS      ;TYPE WITHOUT LEADING ZEROS
6737 027472 104401 043727      TYPE   ,MSGOUS
6738 027476 104401 043720      45:   TYPE   ,MSGAVG      ;"AVG="
6739 027502 012446              MOV    (R4)+, -(SP)     ;FORM THE AVERAGE
6740 027504 012446              MOV    (R4)+, -(SP)
6741 027506 012446              MOV    (R4)+, -(SP)
6742 027510 004737 023614      JSR    PC, @#SDIV
6743 027514 006126              ROL    (SP)+            ;IS THE REMAINDER OVER HALF?
6744 027516 100001              BPL    55               ;NO--BRANCH
6745 027520 005216              INC    (SP)            ;YES--ROUND UP
6746 027522              55:
6747 027522 004737 023222      JSR    PC, @#SSB2D      ;CHANGE TO DECIMAL ASCIZ
6748 027526 004737 023452      JSR    PC, @#SSUPRS      ;TYPE WITHOUT LEADING ZEROS
6749 027532 104401 043727      TYPE   ,MSGOUS
6750 027536 104401 044042      TYPE   ,MSG SP
6751 027542 016446 177776      MOV    -2(R4), -(SP)    ;PUT -2(R4) ON THE STACK
6752 027546 004737 023222      JSR    PC, @#SSB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
6753 027552 004737 023452      JSR    PC, @#SSUPRS      ;TYPE WITHOUT LEADING ZEROS
6754 027556 104401 044012      TYPE   ,MSGNUM
6755 027562 010537 027306      MOV    R5, 25           ;NEXT MESSAGE POINTER
6756 027566 001404              BEQ    75               ;IF NONE EXIT
6757 027570 005005              CLR    R5               ;NO MORE THAN 2
6758 027572 000644              BR     15
6759 027574 104401 044027      65:   TYPE   ,MSGNON
6760 027600 000200      75:   RTS     R0          ;EXIT

; THIS SUBROUTINE WILL INCREMENT THE TRACK
; NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'.
; CALL
;
;       JSR    R0, @#INCTRK
;       RETURN1
;       RETURN2
;       ; TRACK NUMBER GREATER THAN LT15
;       ; TRACK NUMBER INCREMENTED

6769 027602 020237 001520      INCTRK: CMP    R2, @#LT      ;LAST TRACK COMPLETED?
6770 027606 001410              BEQ    25               ;YES--EXIT
6771 027610 063702 001522      ADD    @#IT, R2         ;NO--UPDATE TRACK
6772 027614 020237 001520      CMP    R2, @#LT
6773 027620 003402              BLE    15               ;NO--EXIT
6774 027622 013702 001520      MOV    @#LT, R2        ;YES--SET TRACK TO LAST TRACK
6775 027626 005720      15:   TST    (R0)+        ;ADJUST FOR RETURN 2
6776 027630 000200      25:   RTS     R0          ;RETURN

; THIS SUBROUTINE WILL INCREMENT THE CYLINDER
; NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'.
; CALL
;
;       JSR    R0, @#INCCYL
;       RETURN1
;       RETURN2
;       ; CYLINDER NUMBER GREATER THAN LC15
;       ; CYLINDER NUMBER INCREMENTED

6786 027632 020137 001512      INCCYL: CMP    R1, @#LC
6787 027636 001410              BEQ    25               ;YES--EXIT
  
```



```

6788 027640 063701 001514          ADD    @#IC,R1      ;NO--UPDATE CYLINDER
6789 027644 020137 001512          CMP    R1,@#LC     ;CYLINDER TO BIG?
6790 027650 003402                BLE    1$          ;NO--EXIT
6791 027652 013701 001512          MOV    @#LC,R1     ;YES--SET CYLINDER TO LAST CYLINDER
6792 027656 005720                1$:   TST    (R0)+   ;ADJUST FOR RETURN 2
6793 027660 000200                2$:   RTS     R0     ;RETURN
6794
6795          ;THIS ROUTINE DECREASES THE SECTOR ADDRESS.
6796          ;CALL
6797          ;
6798          ;   CLR    -(SP)      ;CLEAR THE STACK
6799          ;   JSR    PC,DECSEC  ;SUBROUTINE ENTRY
6800          ;   RETURN
6801 027662 113766 004212 000002  DECSEC: MOV    RP,REG+RPDA,2(SP) ;PUT THE SECTOR ADDRESS ON THE STACK
6802 027670 005366 000002          DEC    2(SP)      ;DECREMENT THE ADDRESS
6803 027674 100003                BPL    1$          ;BR IF NOT CORRECTION NEEDED
6804 027676 013766 001634 000002  1$:   MOV    PRMLMT+22,-2(SP) ;OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
6805 027704 000207                RTS     PC         ;RETURN
6806
6807          ;THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
6808          ;WITH ADDRESSES FROM 0 TO 21 WITH EACH ADDRESS
6809          ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
6810          ;CALL
6811          ;
6812          ;   JSR    PC,@#FILBUF
6813          ;   RETURN
6814 027706 104412                FILBUF: SAVREG          ;SAVE R0 - R5
6815 027710 005000                CLR    R0          ;FIRST DISK ADDRESS
6816 027712 012701 047714          MOV    #BUFFER,R1  ;START FILLING HERE
6817 027716 012702 000400  1$:   MOV    #256,R2    ;DO 256 WORDS
6818 027722 010021                2$:   MOV    R0,(R1)+ ;STORE
6819 027724 005302                DEC    R2          ;MORE?
6820 027726 003375                BGT    2$          ;YES--BRANCH
6821 027730 005200                INC    R0          ;NO--UPDATE DISK ADDRESS
6822 027732 023700 001630          CMP    PRMLMT+22,R0 ;DONE?
6823 027736 103367                BHIS  1$          ;NO--BRANCH
6824 027740 104413                RESREG          ;RESTORE R0 - R5
6825 027742 000207                RTS     PC         ;RETURN
6826
6827          ;THIS ROUTINE WILL CLEAR THE BUFFER BY
6828          ;SETTING EACH WORD TO "177400".
6829          ;CALL
6830          ;
6831          ;   JSR    R0,@#CLRBUF
6832          ;   RETURN
6833 027744 104412                CLRBUF: SAVREG          ;SAVE R0 - R5
6834 027746 012701 177400          MOV    #177400,R1  ;WORD TO FILL BUFFER WITH
6835 027752 012702 047714          MOV    #BUFFER,R2  ;FIRST ADDRESS OF BUFFER
6836 027756 012703 075714          MOV    #BUFFER+(512.*22),R3 ;LAST ADDRESS+2 OF BUFFER
6837 027762 010122  1$:   MOV    R1,(R2)+   ;FILL WORDS 1, 9,... 249,... 5625
6838 027764 010122          MOV    R1,(R2)+   ;FILL WORDS 2,10,... 250,... 5626
6839 027766 010122          MOV    R1,(R2)+   ;FILL WORDS 3,11,... 251,... 5627
6840 027770 010122          MOV    R1,(R2)+   ;FILL WORDS 4,12,... 252,... 5628
6841 027772 010122          MOV    R1,(R2)+   ;FILL WORDS 5,13,... 253,... 5629
6842 027774 010122          MOV    R1,(R2)+   ;FILL WORDS 6,14,... 254,... 5630
6843 027776 010122          MOV    R1,(R2)+   ;FILL WORDS 7,15,... 255,... 5631
    
```

```

6844 030000 010122          MOV     R1,(R2)+      ;FILL WORDS 8,16,...256,...5632
6845 030002 020203          CMP     R2,R3        ;DONE?
6846 030004 103766          BLO    15            ;NO--BRANCH
6847 030006 104413          RESREG ;RESTORE R0 - R5
6848 030010 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 030012 104412          CKSCTR: SAVREG      ;SAVE R0 - R5
6858 030014 162706 000004      SUB     #4,SP        ;RESERVE TEMP. STORAGE AREA
6859 030020 005001          CLR     R1           ;FIRST SECTOR
6860 030022 012716 047714      MOV     #BUFFER,(SP) ;FIRST ADDRESS OF DATA BUFFER
6861 030026 005066 000002      CLR     2(SP)        ;NO ERRORS
6862 030032 012702 000020 15:  MOV     #16,R2 ;LOOP COUNT (16*16=256)
6863 030036 011603          MOV     (SP),R3      ;GET 1ST ADDRESS OF THIS SECTORS DATA
6864 030040 25:
6865 030040 020123          CMP     R1,(R3)+     ;WORD 1
6866 030042 001063          BNE    7$           ;BRANCH IF BAD
6867 030044 020123          CMP     R1,(R3)+     ;WORD 2
6868 030046 001061          BNE    7$           ;BRANCH IF BAD
6869 030050 020123          CMP     R1,(R3)+     ;WORD 3
6870 030052 001057          BNE    7$           ;BRANCH IF BAD
6871 030054 020123          CMP     R1,(R3)+     ;WORD 4
6872 030056 001055          BNE    7$           ;BRANCH IF BAD
6873 030060 020123          CMP     R1,(R3)+     ;WORD 5
6874 030062 001053          BNE    7$           ;BRANCH IF BAD
6875 030064 020123          CMP     R1,(R3)+     ;WORD 6
6876 030066 001051          BNE    7$           ;BRANCH IF BAD
6877 030070 020123          CMP     R1,(R3)+     ;WORD 7
6878 030072 001047          BNE    7$           ;BRANCH IF BAD
6879 030074 020123          CMP     R1,(R3)+     ;WORD 8
6880 030076 001045          BNE    7$           ;BRANCH IF BAD
6881 030100 020123          CMP     R1,(R3)+     ;WORD 9
6882 030102 001043          BNE    7$           ;BRANCH IF BAD
6883 030104 020123          CMP     R1,(R3)+     ;WORD 10
6884 030106 001041          BNE    7$           ;BRANCH IF BAD
6885 030110 020123          CMP     R1,(R3)+     ;WORD 11
6886 030112 001037          BNE    7$           ;BRANCH IF BAD
6887 030114 020123          CMP     R1,(R3)+     ;WORD 12
6888 030116 001035          BNE    7$           ;BRANCH IF BAD
6889 030120 020123          CMP     R1,(R3)+     ;WORD 13
6890 030122 001033          BNE    7$           ;BRANCH IF BAD
6891 030124 020123          CMP     R1,(R3)+     ;WORD 14
6892 030126 001031          BNE    7$           ;BRANCH IF BAD
6893 030130 020123          CMP     R1,(R3)+     ;WORD 15
6894 030132 001027          BNE    7$           ;BRANCH IF BAD
6895 030134 020123          CMP     R1,(R3)+     ;WORD 16
6896 030136 001025          BNE    7$           ;BRANCH IF BAD
6897 030140 005302          DEC     R2           ;FINISHED WITH THIS SECTORS DATA?
6898 030142 001336          BNE    2$           ;NO--BRANCH
6899 030144 062716 001000 35:  ADD     #512,(SP)    ;YES--FIRST ADDRESS OF NEXT SECTOR

```



```

6900 030150 005201          INC      R1          ;MOVE TO NEXT SECTOR
6901 030152 023701 001630  CMP      PRMLHT+22, R1 ;DONE?
6902 030156 103325          BHIS     15          ;NO--BRANCH
6903 030160 005766 000002 45:    TST      2(SP)        ;ERROR OCCUR?
6904 030164 001406          BEQ      65          ;NO--BRANCH
6905 030166 123737 001364 001103  CMPB    @#ERR. CT, @#SERFLG ;MAX. ERROR OCCURRED?
6906 030174 101002          BH!     65          ;NO--BRANCH
6907 030176 013700 001252 55:    MOV      @#BYPASS, R0   ;TAKE ERROR EXIT
6908 030202 062706 000004 65:    ADD      #4, SP        ;FREE TEMP. AREA
6909 030206 104413          RESREG          ;RESTORE R0 - R5
6910 030210 000200          RTS     R0          ;RETURN FROM CALL
6911 030212 010304          75:    MOV      R3, R4        ;FORM WORD NUMBER AND
6912 030214 161604          SUB     (SP), R4      ;ADDRESS TO CONTINUE FROM
6913 030216 010405          MOV     R4, R5
6914 030220 706204          ASR     R4          ;WORD NUMBER
6915 030222 042705 177740          BIC     # C37, R5
6916 030226 001002          BNE     85          ;BRANCH IF NOT A MULTIPLE OF 16
6917 030230 012705 000040          MOV     #40, R5      ;SET TO WORD 16
6918 030234 006305          85:    ASL     R5
6919 030236 062705 030040          ADD     #25, R5      ;ADDRESS
6920 030242 016337 177776 001126  MOV     -2(R3), @#SDDAT ;SAVE BAD DATA
6921 030250 005766 000002          TST     2(SP)        ;FIRST ERROR?
6922 030254 001015          BNE     105         ;NO--BRANCH
6923 030256 013737 004176 001270  MOV     @#DTADPB+12, @#CYL. DS ;CYLINDER NUMBER
6924 030264 113737 004175 001274  MOVB    @#DTADPB+11, @#TRK. DS ;TRACK NUMBER
6925 030272 012737 030302 001206  MOV     #95, $ESCAPE  ;;ESCAPE TO 95 ON ERROR
6926 030300 104021          ERROR  21          ;REPORT THE ERROR
6927 030302 105166 000002 95:    COMB    2(SP)        ;SET ERROR SWITCH
6928 030306 000404          BR      115
6929 030310          105:
6930 030310 012737 030320 001206  MOV     #115, $ESCAPE ;;ESCAPE TO 115 ON ERROR
6931 030316 104022          ERROR  22          ;REPORT THE ERROR
6932 030320 032777 001000 150612 115:  BIT     #SW09, @SWR   ;LOOP ON ERROR?
6933 030326 001323          BNE     55          ;YES
6934 030330 032777 000002 150602  BIT     #SW01, @SWR   ;STOP DATA COMPARE?
6935 030336 001310          BNE     45          ;YES--BRANCH
6936 030340 123737 001364 001103  CMPB    @#ERR. CT, @#SERFLG ;MAX. ERRORS?
6937 030346 101713          BLOS   55          ;YES--BRANCH
6938 030350 032777 000040 150562  BIT     #SW05, @SWR   ;REPORT ONLY 1ST ERROR PER SECTOR?
6939 030356 001272          BNE     35          ;YES--BRANCH
6940 030360 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 SETBUF: SAVREG          ; SAVE R0 - R5
6949 MOV     #BUFFER, R1        ; FIRST ADDRESS
6950 MOV     @#DTADPB+4, R2     ; WORD COUNT
6951 15:    MOV     PAT. PT(R0), R3   ; PICKUP PATTERN POINTER
6952 MOV     (R3)+, (R1)+      ; MOVE WORD 1 INTO DATA BUFFER
6953 MOV     (R3)+, (R1)+      ; MOVE WORD 2 INTO DATA BUFFER
6954 MOV     (R3)+, (R1)+      ; MOVE WORD 3 INTO DATA BUFFER
6955 MOV     (R3)+, (R1)+      ; MOVE WORD 4 INTO DATA BUFFER

```



6956	030410	012321		MOV	(R3)+, (R1)+	; MOVE WORD 5 INTO DATA BUFFER
6957	030412	012321		MOV	(R3)+, (R1)+	; MOVE WORD 6 INTO DATA BUFFER
6958	030414	012321		MOV	(R3)+, (R1)+	; MOVE WORD 7 INTO DATA BUFFER
6959	030416	012321		MOV	(R3)+, (R1)+	; MOVE WORD 8 INTO DATA BUFFER
6960	030420	012321		MOV	(R3)+, (R1)+	; MOVE WORD 9 INTO DATA BUFFER
6961	030422	012321		MOV	(R3)+, (R1)+	; MOVE WORD 10 INTO DATA BUFFER
6962	030424	012321		MOV	(R3)+, (R1)+	; MOVE WORD 11 INTO DATA BUFFER
6963	030426	012321		MOV	(R3)+, (R1)+	; MOVE WORD 12 INTO DATA BUFFER
6964	030430	012321		MOV	(R3)+, (R1)+	; MOVE WORD 13 INTO DATA BUFFER
6965	030432	012321		MOV	(R3)+, (R1)+	; MOVE WORD 14 INTO DATA BUFFER
6966	030434	012321		MOV	(R3)+, (R1)+	; MOVE WORD 15 INTO DATA BUFFER
6967	030436	012321		MOV	(R3)+, (R1)+	; MOVE WORD 16 INTO DATA BUFFER
6968	030440	062702	000020	ADD	#16, R2 ; DONE?	
6969	030444	001353		BNE	15	; NO--BRANCH
6970	030446	104413		RESREG		; RESTORE R0 - R5
6971	030450	000207		RTS	PC	; RETURN
6972						
6973						
6974						
6975						
6976						
6977						
6978						
6979						
6980	030452	104412				
6981	030454	012701	047714			
6982	030460	013702	004170			
6983	030464	005046				
6984	030466	016003	003044			
6985	030472					
6986	030472	162321				
6987	030474	001044				
6988	030476	162321				
6989	030500	001042				
6990	030502	162321				
6991	030504	001040				
6992	030506	162321				
6993	030510	001036				
6994	030512	162321				
6995	030514	001034				
6996	030516	162321				
6997	030520	001032				
6998	030522	162321				
6999	030524	001030				
7000	030526	162321				
7001	030530	001026				
7002	030532	162321				
7003	030534	001024				
7004	030536	162321				
7005	030540	001022				
7006	030542	162321				
7007	030544	001020				
7008	030546	162321				
7009	030550	001016				
7010	030552	162321				
7011	030554	001014				

; THIS ROUTINE COMPARES A 16 WORD DATA PATTERN  
 ; AGAINST THE DATA BUFFER

; CALL

;       MOV       #NX, R0               ; PATTERN NUMBER INDEX TO R0  
 ;       JSR       PC, @#DATCMP  
 ;       RETURN

DATCMP: SAVREG                       ; SAVE R0 - R5  
 MOV       #BUFFER, R1               ; FIRST ADDRESS OF BUFFER  
 MOV       @#DTADPB+4, R2           ; WORD COUNT  
 CLR       -(SP)                     ; NO ERROR  
 15: MOV       PAT. PT(R0), R3       ; PATTERN POINTER

25:  
 SUB       (R3)+, (R1)+             ; CHECK WORD 1  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 2  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 3  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 4  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 5  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 6  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 7  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 8  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 9  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 10  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 11  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 12  
 BNE       45                       ; BRANCH IF DIFFERENT  
 SUB       (R3)+, (R1)+             ; CHECK WORD 13  
 BNE       45                       ; BRANCH IF DIFFERENT



```

7012 030556 162321 SUB (R3)+, (R1)+ ;CHECK WORD 14
7013 030560 001012 BNE 45 ;BRANCH IF DIFFERENT
7014 030562 162321 SUB (R3)+, (R1)+ ;CHECK WORD 15
7015 030564 001010 BNE 45 ;BRANCH IF DIFFERENT
7016 030566 162321 SUB (R3)+, (R1)+ ;CHECK WORD 16
7017 030570 001006 BNE 45 ;BRANCH IF DIFFERENT
7018 030572 062702 000020 ADD #16, R2 ;DONE ?
7019 030576 001333 BNE 15 ;NO--BRANCH
7020 030600 005726 35: TST (SP)+ ;YES -- CLEAN UP STACK
7021 030602 104413 RESREG ;RESTORE R0 - R5
7022 030604 000207 RTS PC
7023 030606 010104 45: MOV R1, R4 ;FORM THE WORD NUMBER
7024 030610 162704 047714 SUB #BUFFER, R4
7025 030614 006204 ASR R4 ;WORD NUMBER
7026 030616 010305 MOV R3, R5 ;FORM ADDRESS TO CONTINUE FROM
7027 030620 166005 003044 SUB PAT. PT(R0), R5
7028 030624 006305 ASL R5
7029 030626 062705 030472 ADD #25, R5 ;ADDRESS
7030 030632 064341 ADD -(R3), -(R1) ;RECONSTRUCT THE BAD WORD
7031 030634 010137 001122 MOV R1, @%SDADR ;SAVE THE ERROR INFORMATION
7032 030640 010337 001120 MOV R3, @%SGADR
7033 030644 012137 001126 MOV (R1)+, @%SDDAT
7034 030650 012337 001124 MOV (R3)+, @%SGDAT
7035 030654 005716 TST (SP) ;1ST DATA COMPARE ERROR?
7036 030656 001023 BNE 65 ;NO--BRANCH
7037 030660 013737 004176 001270 MOV @%DTADPB+12, @%CYL. DS ;CYLINDER
7038 030666 113737 004175 001274 MOV @%DTADPB+11, @%TRK. DS ;TRACK
7039 030674 113737 004174 001272 MOV @%DTADPB+10, @%SEC. DS ;SECTOR
7040 030702 016600 000026 MOV 26(SP), R0 ;GET TEST PC+4
7041 030706 012737 030716 001206 MOV #55, %ESCAPE ;ESCAPE TO 55 ON ERROR
7042 030714 104013 ERROR 13 ;REPORT THE ERROR
7043 030716 016600 000020 55: MOV 20(SP), R0 ;PATTERN NUMBER INDEX
7044 030722 105116 COMB (SP) ;SET THE ERROR SWITCH
7045 030724 000404 BR 75
7046 030726 65:
7047 030726 012737 030736 001206 MOV #75, %ESCAPE ;ESCAPE TO 75 ON ERROR
7048 030734 104014 ERROR 14 ;REPORT THE ERROR
7049 030736 032777 000002 150174 75: BIT #SW01, %SWR ;STOP DATA COMPARE?
7050 030744 001315 BNE 35 ;YES--EXIT
7051 030746 123737 001364 001103 CMPB @%ERR. CT, @%SERFLG ;MAX. ERRORS?
7052 030754 101004 BHI 85 ;NO--BRANCH
7053 030756 013766 001252 000016 MOV @%BYPASS, 16(SP) ;YES--ERROR EXIT
7054 030764 000705 BR 35
7055 030766 000115 85: 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 030770 012701 047714 FILRAN: MOV #BUFFER, R1
7067 030774 013702 001630 MOV PRMLM+22, R2 ;MAXIMUM NUMBER OF SECTORS
  
```



```

7068 031000 004037 031210      15:   JSR   RO, @RANPAT
7069 031004 005302              DEC   R2
7070 031006 100374              BPL   15
7071 031010 000200              RTS   RO
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   RO, @RANCK
7079                               ;   RETURN
7080
7081 031012 013746 023610      RANCK:  MOV   @%SHINUM - (SP) ; SAVE THE PRESENT RANDOM NUMBER
7082 031016 013746 023612      MOV   @%SLONUM - (SP)
7083 031022 012702 050714      MOV   @BUFFER+512, R2 ; READ BUFFER ADDRESS
7084 031026 012701 051714      MOV   @BUFFER+1024, R1 ; RANDOM PATTERN ADDRESS
7085 031032 010103              MOV   R1, R3 ; COPY IT INTO R3 FOR LATER USE
7086 031034 011237 023612      MOV   (R2), @%SLONUM ; PRIME THE RANDOM NUMBER GENERATOR
7087 031040 016237 000002 023610  MOV   2(R2), @%SHINUM
7088 031046 004037 031210      JSR   RO, @RANPAT ; GENERATE A RANDOM PATTERN
7089 031052 012637 023612      MOV   (SP)+, @%SLONUM ; RESTORE PRESENT RANDOM NUMBER
7090 031056 012637 023610      MOV   (SP)+, @%SHINUM
7091 031062 005046              CLR   -(SP) ; NO ERRORS
7092 031064 162322              15:   SUB   (R3)+, (R2)+ ; ARE THESE TWO WORDS DIFFERENT?
7093 031066 001441              BEQ   45 ; NO--BRANCH
7094 031070 012737 031142 001206  MOV   #35, %$ESCAPE ; ESCAPE TO 35 ON ERROR
7095 031076 064342              ADD   -(R3), -(R2) ; RECREATE THE BAD WORD
7096 031100 010237 001122      MOV   R2, @%$BDADR ; ADDRESS OF BAD DATA
7097 031104 010337 001120      MOV   R3, @%$GDADR ; ADDRESS OF GOOD DATA
7098 031110 012237 001126      MOV   (R2)+, @%$BDDAT ; BAD DATA
7099 031114 012337 001124      MOV   (R3)+, @%$GDDAT ; GOOD DATA
7100 031120 010204              MOV   R2, R4 ; FORM WORD NUMBER (1 TO 256)
7101 031122 162704 050714      SUB   @BUFFER+512, R4
7102 031126 006204              ASR   R4
7103 031130 005716              TST   (SP) ; FIRST ERROR
7104 031132 001002              BNE   25 ; NO--BRANCH
7105 031134 105116              COMB (SP) ; YES--SET ERROR SWITCH
7106 031136 104015              ERROR 15 ; REPORT THE ERROR
7107 031140 104016              25:  ERROR 16 ; REPORT THE ERROR
7108 031142 032777 001000 147770  35:  BIT   @SW09, @SWR ; LOOP ON ERROR?
7109 031150 001012              BNE   55 ; YES--BRANCH
7110 031152 123737 001364 001103      CMPB @%ERR. CT, @%$ERFLG ; MAX. ERRORS OCCURRED?
7111 031160 101406              BLOS  55 ; YES--BRANCH
7112 031162 032777 000002 147750      BIT   @SW01, @SWR ; STOP COMPARING?
7113 031170 001002              BNE   55 ; YES--BRANCH
7114 031172 020103              45:  CMP   R1, R3 ; ALL DATA BEEN COMPARED?
7115 031174 101333              BHI   15 ; NO--BRANCH
7116 031176 005726              55:  TST   (SP)+ ; ERROR OCCUR?
7117 031200 001402              BEQ   65 ; NO--BRANCH
7118 031202 013700 001252      MOV   @%BYPASS, RO ; TAKE ERROR EXIT
7119 031206 000200              65:  RTS   RO ; 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      RO,@#RANPAT
7127 ;      RETURN
7128
7129 RANPAT: MOV      R2,-(SP)      ;SAVE R2
7130 031210 010246 000200      MOV      #256./2.,R2      ;GENERATE 256 WORDS
7131 031216 000402          BR      25
7132 031220 004737 023512      15:     JSR      PC,@#SRAND      ;GENERATE A RANDOM NUMBER
7133 031224 013721 023612      25:     MOV      @#SLONUM(R1)+ ;PUT LOW WORD IN BUFFER
7134 031230 013721 023610      MOV      @#SHINUM(R1)+ ;PUT HIGH WORD IN BUFFER
7135 031234 005302          DEC      R2      ;DONE?
7136 031236 003370          BGT      15      ;NO--BRANCH
7137 031240 012602          MOV      (SP)+,R2      ;RESTORE R2
7138 031242 000200          RTS      RO      ;EXIT
7139
7140 ; THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
7141 ; ADDRESSES AND SAVES THEM IN THE DPB (DTADPB+10 AND DTADPB+12).
7142 ; NOTE: THIS ROUTINE DESTROYS R1-R3
7143 ; CALL
7144 ;      JSR      RO,@#RANADR
7145 ;      RETURN
7146
7147 RANADR: JSR      PC,@#SRAND      ;GENERATE A RANDOM NUMBER
7148 031244 004737 023512      MOV      @#SLONUM,R1      ;FORM SECTOR IN R1
7149 031250 113701 023612      MOV      @#SLONUM+1,R2    ;FORM TRACK IN R2
7150 031254 113702 023613      MOV      @#SHINUM,R3     ;FORM CYLINDER IN R3
7151 031260 013703 023610      TSTB   R1      ;ENSURE THE SECTOR IS BETWEEN 0 AND 21
7152 031264 105701          BLT     25
7153 031270 123701 001630      15:     CMPB   PRMLT+22,R1      ;CHECK MAXIMUM SECTOR ADDRESS
7154 031274 103003          BHIS   35
7155 031276 000241          CLC
7156 031300 106001          RORB   R1
7157 031302 000772          BR     15
7158 031304 105702          35:     TSTB   R2      ;ENSURE THE TRACK IS BETWEEN 0 AND 18
7159 031306 002403          BLT     55
7160 031310 122702 000023      45:     CMPB   #19.,R2
7161 031314 003003          BGT     65
7162 031316 000241          55:     CLC
7163 031320 106002          RORB   R2
7164 031322 000772          BR     45
7165 031324 023703 001510      65:     CMP    @#FC,R3      ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
7166 031330 003413          BLE    75
7167 031332 000241          CLC
7168 031334 006003          ROR    R3
7169 031336 005503          ADC    R3
7170 031340 001371          BNE    65
7171 031342 010103          MOV    R1,R3
7172 031344 000303          SWAB  R3
7173 031346 060203          ADD    R2,R3
7174 031350 005203          INC    R3
7175 031352 003364          BGT    65
7176 031354 005403          NEG    R3
7177 031356 000762          BR     65
7178 031360 023703 001512      75:     CMP    @#LC,R3
7179 031364 002003          BGE    85
    
```

```

7180 031366 000241          CLC
7181 031370 006003          ROR      R3
7182 031372 000772          BR       75
7183 031374 023703 001510 85:  CMP     @#FC,R3
7184 031400 003403          BLE     95
7185 031402 005203          INC     R3
7186 031404 000303          SWAB   R3
7187 031406 000764          BR       75
7188 031410 110137 004174 95:  MOVB   R1,@#DTADPB+10 ;SAVE SECTOR ADDRESS
7189 031414 110237 004175    MOVB   R2,@#DTADPB+11 ;SAVE TRACK ADDRESS
7190 031420 010337 004176    MOV    R3,@#DTADPB+12 ;SAVE CYLINDER ADDRESS
7191 031424 000200          RTS     RO ;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 031426 032777 000200 147504 GETSWR: BIT     #SW07,@SWR ;READ CONTROL SWITCHES?
7202 031434 001430          BEQ     25 ;NO--BRANCH
7203 031436 104401 031444    TYPE   ,655 ;TYPE ASCIZ STRING
7204 031442 000410          BR     645 ;GET OVER THE ASCIZ
7205 ; ; 655: .ASCIZ <CR><LF>/SET SWR<07>=0/
7206 031464
7207 031464 012703 043052 645:  MOV    #MSG.CS,R3 ;"CONTROL SWITCHES="
7208 031470 013704 001220    MOV    @#C.SWR,R4 ;PRESENT CONTROL SWITCH SETTINGS
7209 031474 004037 031520    JSR   RO,@#GETNUM ;GET THE NEW SWITCH SETTINGS
7210 031500 000771          BR     15 ;COMMA
7211 031502 000240          NOP    ;PERIOD
7212 031504 013737 001220 001222    MOV    C.SWR,SAVCSW ;SAVE PREVIOUS VALUE
7213 031512 010437 001220    MOV    R4,@#C.SWR ;DOUBLE PERIOD--SAVE NEW SWITCH SETTING
7214 031516 000207 25:  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   RO,@#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 031520 010337 031526    GETNUM: MOV    R3,25 ;SAVE MESSAGE POINTER
7232 031524 104401 15:  TYPE   ;TYPE THE MESSAGE
7233 031526 000000 25:  .WORD 0 ;MESSAGE POINTER GOES HERE
7234 031530 010446    MOV    R4,-(SP) ;SAVE R4 FOR TYPEOUT
7235 031532 104402    TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)

```



```

7236 031534 104401 043076          TYPE      ,SLASH          ;' / '
7237 031540 104411          RDLIN          ;READ AN ASCIZ STRING
7238 031542 012601          MOV          (SP)+,R1      ;ADDRESS OF FIRST CHARACTER
7239 031544 004037 033770      JSR          RO,@#CK. CHR  ;CHECK ONE CHARACTER
7240 031550 031524          1$           ;ILLEGAL CHARACTER
7241 031552 031524          1$           ;CARRIAGE RETURN
7242 031554 031566          3$           ;"/"
7243 031556 031612          7$           ;","
7244 031560 031620          8$           ;"."
7245 031562 031564          11$          ;DIGIT 0-9
7246 031564 005301          11$: DEC      R1          ;DECREMENT THE INPUT POINTER
7247 031566
7248 031566 004037 034230      JSR          RO,@#CK. NUM  ;CHECK THE NUMBER
7249 031572 031524          1$           ;ILLEGAL INPUT
7250 031574 031606          6$           ;TERMINATED WITH A "," OR "CR"
7251 031576 031604          5$           ;TERMINATED WITH A "."
7252 031600 031602          4$           ;TERMINATED WITH A ".."
7253 031602 005720          4$: TST      (RO)+       ;DOUBLE PERIOD
7254 031604 005720          5$: TST      (RO)+       ;SINGLE PERIOD
7255 031606 010204          6$: MOV      R2,R4        ;COMMA--SAVE INPUT NUMBER
7256 031610 000414          BR          10$         ;GO TO EXIT
7257 031612 105711          7$: TSTB   (R1)         ;TERMINATOR AFTER A COMMA?
7258 031614 001343          BNE       1$           ;NO--LOOP
7259 031616 000411          BR          10$         ;YES--EXIT
7260 031620 105711          8$: TSTB   (R1)         ;TERMINATOR AFTER A PERIOD?
7261 031622 001406          BEQ      9$           ;YES--EXIT
7262 031624 122721 000056      CMPB   #'',(R1)+       ;NO--DOUBLE PERIOD?
7263 031630 001335          BNE       1$           ;NO--LOOP
7264 031632 105711          TSTB   (R1)         ;YES--TERMINATOR?
7265 031634 001333          BNE       1$           ;NO--LOOP
7266 031636 005720          TST      (RO)+       ;DOUBLE PERIOD
7267 031640 005720          9$: TST      (RO)+       ;PERIOD
7268 031642 010402          10$: MOV     R4,R2       ;COMMA--POSITION THE
7269 031644 000302          SWAB    R2           ;NUMBER IN CASE IT
7270 031646 006202          ASR     R2           ;IS THE PRIORITY LEVEL
7271 031650 006202          ASR     R2
7272 031652 006202          ASR     R2
7273 031654 000200          RTS      RO          ;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 031656 104412          GT. PRM: SAVREG        ;SAVE R0 - R5
7282 031660 005037 001232      GT. PR1: CLR          DRVSEL ;NO DRIVE SELECTED
7283 031664 104401 031672      TYPE      ,65$        ;TYPE ASCIZ STRING
7284 031670 000406          BR          64$        ;GET OVER THE ASCIZ
7285          ; 65$: .ASCIZ <CR><LF>/DRIVE(S)=/
7286 031706
7287 031706 104411          64$: RDLIN          ;READ TTY
7288 031710 012601          MOV      (SP)+,R1      ;ADDRESS OF ASCIZ STRING
7289 031712 004037 033770      JSR      RO,@#CK. CHR  ;CHECK ONE CHARACTER
7290 031716 031660          GT. PR1          ;ILLEGAL CHARACTER
7291 031720 031660          GT. PR1          ;CARRIAGE RETURN
  
```

```

7292 031722 031660          GT. PR1          ;"/"
7293 031724 031660          GT. PR1          ;"."
7294 031726 031660          GT. PR1          ;"."
7295 031730 031732          15          ;DIGIT 0-9
7296 031732 005301          15          DEC      R1
7297 031734          25:
7298 031734 012702 000007    MOV      #7,R2 ;UPPER LIMIT OF INPUT
7299 031740 004037 034044    JSR      RO, @ACK.DIG ;CHECK THE DIGIT(S)
7300 031744 031660          GT. PR1          ;ILLEGAL INPUT
7301 031746 031660          GT. PR1          ;INPUT TO LARGE
7302 031750 031756          35          ;TERMINATED WITH A ",", " OR "CR"
7303 031752 032002          45          ;TERMINATED WITH A "."
7304 031754 032002          45          ;TERMINATED WITH A " "
7305 031756 156237 034504 001232 35:  BISB    ATABIT(R2), DRVSEL ;SET THE DRIVE SELECTED BIT
7306 031764 105741          TSTB    -(R1) ;WAS THE LINE TERMINATED?
7307 031766 001362          BNE     25          ;NO-GET THE NEXT DRIVE
7308 031770 005037 001234    CLR     @TSTNMS ;DESELECT ALL TESTS
7309 031774 005037 001236    CLR     TSTNMS+2
7310 032000 000405          BR      GTTST1 ;YES--SELECT TEST
7311 032002 156237 034504 001232 45:  BISB    ATABIT(R2), DRVSEL ;SET THE SELECTED DRIVE BITS
7312 032010 104413          GT. PR2: RESREG ;RESTORE RO - R5
7313 032012 000207          RTS     PC ;EXIT
7314
7315 032014          GTTST1:
7316 032014 104401 032022    TYPE    ,655 ;;TYPE ASCIZ STRING
7317 032020 000403          BR      645 ;;GET OVER THE ASCIZ
7318          ;;655: .ASCIZ /TEST=/
7319          645:
7320 032030 104411          RDLIN ;READ AN ASCIZ STRING
7321 032032 012601          MOV     (SP)+, R1 ;POINTER TO R1
7322 032034 122711 000056    CMPB   #'.,(R1) ;DOUBLE PERIOD?
7323 032040 001007          BNE    15 ;NO--BRANCH
7324 032042 122761 000056 000001  CMPB   #'.,1(R1)
7325 032050 001003          BNE    15
7326 032052 105761 000002    TSTB   2(R1) ;"CR"?
7327 032056 001754          BEQ    GT. PR2 ;YES--EXIT
7328 032060 005037 001234 15:  CLR     TSTNMS ;NO TEST SELECTED
7329 032064 005037 001236    CLR     TSTNMS+2
7330 032070 005037 001240    CLR     OPNFLG ;NO TESTS TO BE OPENED
7331 032074 005037 001242    CLR     OPNFLG+2
7332 032100 121127 000123  GTTST2: CMPB   (R1), #'S ;ALL SEEK TESTS?
7333 032104 001004          BNE    15 ;NO--BRANCH
7334 032106 052737 000777 001234  BIS    #777, TSTNMS ;YES--SELECT TESTS 0-10
7335 032114 000552          BR     GTTST3
7336 032116 121127 000124 15:  CMPB   (R1), #'T ;ALL TIMING TESTS?
7337 032122 001004          BNE    25 ;NO--BRANCH
7338 032124 052737 036000 001234  BIS    #36000, TSTNMS ;YES--SELECT TESTS 12-15
7339 032132 000543          BR     GTTST3
7340 032134 121127 000101 25:  CMPB   (R1), #'A ;ALL ADDRESSING TESTS?
7341 032140 001004          BNE    35 ;NO--BRANCH
7342 032142 052737 140000 001234  BIS    #140000, TSTNMS ;YES--SELECT TESTS 16 & 17
7343 032150 000534          BR     GTTST3
7344 032152 121127 000104 35:  CMPB   (R1), #'D ;DATA TEST?
7345 032156 001004          BNE    45 ;NO--BRANCH
7346 032160 052737 000001 001236  BIS    #1, TSTNMS+2 ;YES--SELECT TEST 20
7347 032166 000525          BR     GTTST3
    
```



7348	032170	121127	000105		45:	CMPB	(R1), #'E	; EXERCISER TEST?
7349	032174	001004				BNE	55	; NO--BRANCH
7350	032176	052737	000002	001236		BIS	#2, TSTNMS+2	; YES--SELECT TEST 21
7351	032204	000516				BR	GTTST3	
7352	032206	004037	033714		55:	JSR	RO, @#CK.OCT	; OCTAL DIGIT?
7353	032212	000514				BR	GTTST4	; NO--BRANCH
7354	032214	010205				MOV	R2, R5	; YES--SAVE IT
7355	032216	005201				INC	R1	; MOVE TO NEXT CHARACTER
7356	032220	004037	033714			JSR	RO, @#CK.OCT	; OCTAL DIGIT
7357	032224	000405				BR	65	; NO--BRANCH
7358	032226	005201				INC	R1	; MOVE TO NEXT CHARACTER
7359	032230	006305				ASL	R5	; SCALE HIGH DIGIT
7360	032232	006305				ASL	R5	
7361	032234	006305				ASL	R5	
7362	032236	060502				ADD	R5, R2	; COMBINE HIGH & LOW DIGITS
7363	032240	020227	000022		65:	CMP	R2, #5TN-1	; VALID TEST NUMBER?
7364	032244	003263				BGT	GTTST1	; NO--BRANCH
7365	032246	010237	032440			MOV	R2, 95	; SAVE THE TEST NUMBER
7366	032252	010204				MOV	R2, R4	; CONVERT TEST NUMBER INTO AN INDEX
7367	032254	042704	000017			BIC	#17, R4	; CLEAR UNWANTED BITS
7368	032260	006204				ASR	R4	; SHIFT THE BITS
7369	032262	006204				ASR	R4	; SHIFT THE BITS
7370	032264	006204				ASR	R4	; SHIFT THE BITS
7371	032266	006302				ASL	R2	
7372	032270	056264	001424	001234		BIS	BITS(R2), TSTNMS(R4)	; SELECT TEST
7373	032276	121127	000055			CMPB	(R1), #'-	; TEST STRING?
7374	032302	001060				BNE	GTTST4	; NO--BRANCH
7375	032304	005201				INC	R1	; YES--MOVE TO NEXT CHARACTER
7376	032306	004037	033714			JSR	RO, @#CK.OCT	; OCTAL DIGIT?
7377	032312	000640				BR	GTTST1	; NO--BRANCH
7378	032314	010205				MOV	R2, R5	; YES--SAVE IT
7379	032316	005201				INC	R1	; MOVE TO NEXT CHARACTER
7380	032320	004037	033714			JSR	RO, @#CK.OCT	; OCTAL DIGIT?
7381	032324	000405				BR	75	; NO--BRANCH
7382	032326	005201				INC	R1	; YES--MOVE TO NEXT CHARACTER
7383	032330	006305				ASL	R5	; SCALE HIGH DIGIT
7384	032332	006305				ASL	R5	
7385	032334	006305				ASL	R5	
7386	032336	060502				ADD	R5, R2	; COMBINE HIGH & LOW DIGIT
7387	032340	020227	000022		75:	CMP	R2, #5TN-1	; VALID TEST NUMBER?
7388	032344	003223				BGT	GTTST1	; NO--BRANCH
7389	032346	023702	032440			CMP	95, R2	; IS THE FIRST NUMBER OF THE
7390								; STRING SMALLER THAN THE LAST?
7391	032352	002220				BGE	GTTST1	; NO--BRANCH
7392	032354	010246				MOV	R2, -(SP)	; SAVE ENDING TEST NUMBER
7393	032356	013702	032440			MOV	95, R2	; GET STARTING TEST NUMBER
7394	032362	012637	032440			MOV	(SP)+, 95	; STORE ENDING TEST NUMBER
7395	032366	006337	032440			ASL	95	; SHIFT ENDING TEST NUMBER
7396	032372	006302				ASL	R2	; SHIFT TEST NUMBER
7397	032374	010204			85:	MOV	R2, R4	; COPY TEST NUMBER INTO R4
7398	032376	042704	000037			BIC	#37, R4	; CLEAR LOWER BITS
7399	032402	006204				ASR	R4	; SHIFT THE TEST NUMBER
7400	032404	006204				ASR	R4	; SHIFT THE TEST NUMBER
7401	032406	006204				ASR	R4	; SHIFT THE TEST NUMBER
7402	032410	006204				ASR	R4	; SHIFT THE TEST NUMBER
7403	032412	056264	001424	001234		BIS	BITS(R2), TSTNMS(R4)	; SELECT THE TEST

```

7404 032420 062702 000002 ADD #2,R2 ; INCREMENT THE TEST NUMBER
7405 032424 020237 032440 CMP R2,95 ; SEE IF FINISHED
7406 032430 101761 BLOS 85 ; BR IF NOT
7407 032432 162702 000002 SUB #2,R2 ; CORRECT TEST NUMBER
7408 032436 000402 BR GTTST4 ; CONTINUE
7409 032440 000000 95: .WORD 0 ; STORE TEST NUMBER HERE
7410 032442 005201 GTTST3: INC R1 ; MOVE TO NEXT CHARACTER
7411 032444 121127 000056 GTTST4: CMPB (R1),#' ; "PERIOD"?
7412 032450 001441 BEQ GTTST5 ; YES--BRANCH
7413 032452 005737 001234 TST TSTNMS ; ANY TEST SELECTED THIS CYCLE?
7414 032456 001005 BNE 15 ; BR IF YES
7415 032460 005737 001236 TST TSTNMS+2 ; ANY TEST SELECTED THIS CYCLE ?
7416 032464 001002 BNE 15 ; BR IF YES
7417 032466 000137 032014 JMP GTTST1 ; NO
7418 032472 121127 000057 15: CMPB (R1),# '/' ; "OPEN"?
7419 032476 001004 BNE 25 ; NO--BRANCH
7420 032500 056264 001424 001240 BIS BITS(R2), OPNFLG(R4) ; YES--SET BITS FOR TEST TO OPEN
7421 032506 000405 BR 35
7422 032510 121127 000054 25: CMPB (R1),# ',' ; "COMMA"?
7423 032514 001402 BEQ 35 ; BR IF YES
7424 032516 000137 032014 JMP GTTST1 ; NO
7425 032522 005201 35: INC R1 ; MOVE TO NEXT CHARACTER
7426 032524 105711 TSTB (R1) ; "CR"?
7427 032526 001402 BEQ 45 ; BR IF 'CR'
7428 032530 000137 032100 JMP GTTST2 ; NO--GO GET NEXT CHARACTER
7429 032534 005737 001240 45: TST OPNFLG ; ANY TESTS TO OPEN ?
7430 032540 001042 BNE OPNTST ; BR IF YES
7431 032542 005737 001242 TST OPNFLG+2 ; ANY TESTS TO OPEN ?
7432 032546 001037 BNE OPNTST ; BR IF YES
7433 032550 000137 032014 JMP GTTST1 ; NO--START AGAIN
7434 032554 005201 GTTST5: INC R1 ; MOVE TO NEXT CHARACTER
7435 032556 121127 000056 CMPB (R1),#' ; "PERIOD"?
7436 032562 001414 BEQ GTTST6 ; YES--BRANCH
7437 032564 105711 TSTB (R1) ; "CR"?
7438 032566 001402 BEQ 15 ; YES--BRANCH
7439 032570 000137 032014 JMP GTTST1
7440 032574 005737 001240 15: TST OPNFLG ; ANY TESTS TO OPEN ?
7441 032600 001022 BNE OPNTST ; BR IF YES
7442 032602 005737 001242 TST OPNFLG+2 ; ANY TESTS TO OPEN ?
7443 032606 001017 BNE OPNTST ; BR IF YES
7444 032610 000137 032010 JMP GT. PR2 ; NO--GO START TESTING
7445 032614 005201 GTTST6: INC R1 ; MOVE TO NEXT CHARACTER
7446 032616 105711 TSTB (R1) ; "CR"?
7447 032620 001402 BEQ 15 ; YES--BRANCH
7448 032622 000137 032014 JMP GTTST1 ; NO--GO ASK FOR TEST
7449 032626 005737 001240 15: TST OPNFLG ; ANY TESTS TO OPEN ?
7450 032632 001005 BNE OPNTST ; BR IF YES
7451 032634 005737 001242 TST OPNFLG+2 ; ANY TESTS TO OPEN ?
7452 032640 001002 BNE OPNTST ; BR IF YES
7453 032642 000137 032010 JMP GT. PR2 ; NO--GO START TESTING
7454
7455 ; OPEN THE SELECTED TEST FOR CHANGES
7456
7457 032646 104412 OPNTST: SAVREG ; SAVE R0 - R5
7458 032650 005027 CLR (PC)+ ; START WITH TEST 0
7459 032652 000000 OPN. CT: .WORD 0 ; COUNT STORED HERE
  
```



```

7460 032654 000411          BR      OPN. 2      ;SKIP THE INCREMENT
7461 032656 005237 032652  OPN. 1:  INC      OPN. CT      ;MOVE TO THE NEXT TEST
7462 032662 022737 000022 032652  CMP      #STN-1, OPN. CT ;TEST NUMBER TOO BIG?
7463 032670 002003          BGE      OPN. 2      ;NO--OPEN THE NEXT TEST
7464 032672 104413          RESREG          ;RESTORE R0 - R5
7465 032674 000137 032014          JMP      GTTST1     ;YES--GO ASK FOR MORE TESTS
7466 032700 013705 032652  OPN. 2:  MOV      OPN. CT, R5   ;SETUP TO USE THE
7467 032704 006305          ASL      R5         ;TEST NUMBER AS AN INDEX
7468 032706 013703 032652  MOV      OPN. CT, R3   ;GET INDEX
7469 032712 042703 000017  BIC      #17, R3     ;CLEAR LOWER TEST BITS
7470 032716 006203          ASR      R3         ;SHIFT TEST NUMBER
7471 032720 006203          ASR      R3         ;SHIFT TEST NUMBER
7472 032722 006203          ASR      R3         ;SHIFT TEST NUMBER
7473 032724 036563 001424 001240  BIT      BITS(R5), OPNFLG(R3) ;OPEN THIS TEST?
7474 032732 001751          BEQ      OPN. 1      ;NO--MOVE TO NEXT TEST
7475 032734 104401 032742  TYPE     , 65$      ;TYPE ASCIZ STRING
7476 032740 000404          BR      64$        ;GET OVER THE ASCIZ
7477          ;: 65$: .ASCIZ / TEST /
7478 032752          64$:
7479 032752 013746 032652  MOV      OPN. CT, -(SP) ;: SAVE OPN. CT FOR TYPEOUT
7480          ;: TEST NUMBER
7481 032756 104403          TYPOS          ;: GO TYPE--OCTAL ASCII
7482 032760 002          . BYTE 2          ;: TYPE 2 DIGIT(S)
7483 032761 000          . BYTE 0          ;: SUPPRESS LEADING ZEROS
7484 032762 104401 001215  TYPE     , $CRLF     ;: TYPE "CR" & "LF"
7485 032766 016500 001536  MOV      PRMPT(R5), R0 ;: PICKUP PARAMETER POINTER
7486 032772 011046          MOV      (R0), -(SP) ;: SAVE THE VARIABLE INDICATOR
7487 032774 012702 001504  MOV      #PRM, R2     ;: FIRST ADDRESS OF TABLE
7488 033000 000405          BR      2$         ;
7489 033002 006216          1$: ASR      (SP)      ;: CHECK FOR A VARIABLE
7490 033004 103403          BCS      2$         ;: GO MOVE THIS ONE
7491 033006 001404          BEQ      OPNPRM     ;: DONE
7492 033010 005722          TST      (R2)+      ;: BUMP THE POINTER
7493 033012 000773          BR      1$         ;
7494 033014 012022          2$: MOV      (R0)+, (R2)+ ;: MOVE THIS VARIABLE INTO THE
7495 033016 000771          BR      1$         ;: COMMON AREA
7496 033020 013716 001504  OPNPRM: MOV     @#PRM, (SP) ;: GET THE VARIABLE INDICATOR
7497 033024 005004          CLR      R4         ;: ZERO THE INDEX
7498 033026 006216          1$: ASR      (SP)      ;: CHECK FOR A VARIABLE
7499 033030 103403          BCS      3$         ;: GO GET IT
7500 033032 001772          BEQ      OPNPRM     ;: OUT OF VARIABLES
7501 033034 005724          2$: TST      (R4)+      ;: UPDATE THE INDEX
7502 033036 000773          BR      1$         ;
7503 033040 005764 001606          3$: TST      PRMLMT(R4) ;: IS THE MAX. MAGNITUDE NEG?
7504 033044 100456          BMI      OPNPAT     ;: YES--THEN IT IS THE PATTERN
7505 033046 104401 044042  TYPE     , MSG. SP   ;: TYPE SPACES
7506 033052 016437 001636 033062  MOV      PRMSG(R4), 4$ ;: TYPE THE NAME OF THIS VARIABLE
7507 033060 104401          TYPE          ;
7508 033062 000000          4$: . WORD 0          ;
7509 033064 104401 043050  TYPE     , MSG. EQ   ;: TYPE "="
7510 033070 016446 001506  MOV      RPT(R4), -(SP) ;: PUT RPT(R4) ON THE STACK
7511 033074 004737 023222  JSR      PC, @$$SB2D ;: CHANGE RPT(R4) TO DECIMAL ASCII
7512 033100 004737 023452  JSR      PC, @$$SUPRS ;: TYPE WITHOUT LEADING ZEROS
7513 033104 104401 043076  TYPE     , SLASH     ;: /
7514 033110 104411          RDL IN          ;
7515 033112 012601          MOV      (SP)+, R1  ;: READ AN ASCII STRING
  
```

7516	033114	004037	033770		JSR	RO, @#CK. CHR	;CHECK ONE CHARACTER
7517	033120	033040			3\$		;ILLEGAL CHARACTER
7518	033122	033034			2\$		;CARRIAGE RETURN
7519	033124	033172			8\$		;"/"
7520	033126	033134			5\$		;": "
7521	033130	033142			6\$		;". "
7522	033132	033170			7\$		;DIGIT 0-9
7523	033134	105711		5\$:	TSTB	(R1)	; "CR"?
7524	033136	001340			BNE	3\$	;NO--STAY ON THIS VARIABLE
7525	033140	000735			BR	2\$	;YES--MOVE TO NEXT VARIABLE
7526	033142	105711		6\$:	TSTB	(R1)	;IS THERE A "CR" AFTER THE PERIOD?
7527	033144	001002			BNE	64\$	;NO
7528	033146	000137	033562		JMP	OPN. N2	;YES--GO CLOSE THIS TEST
7529	033152	122721	000056	64\$:	CMPB	#'. , (R1)+	;DOUBLE PERIOD?
7530	033156	001330			BNE	3\$	;NO--GO ASK FOR THIS VARIABLE
7531	033160	105711			TSTB	(R1)	;YES--IS A "CR" AFTER THE DOUBLE PERIOD?
7532	033162	001326			BNE	3\$	;NO--ASK FOR THIS VARIABLE AGAIN
7533	033164	000137	033600		JMP	OPN. X2	;YES--CLOSE ALL TEST
7534	033170	005301		7\$:	DEC	R1	;BACK THE POINTER UP BY ONE
7535	033172			8\$:			
7536	033172	016402	001606		MOV	PRMLT(R4), R2	;UPPER LIMIT OF INPUT
7537	033176	004037	034044		JSR	RO, @#CK. DIG	;CHECK THE DIGIT(S)
7538	033202	033040			3\$		;ILLEGAL INPUT
7539	033204	033040			3\$		;INPUT TOO LARGE
7540	033206	033214			9\$		;TERMINATED WITH A ", " OR "CR"
7541	033210	033556			OPN. N1		;TERMINATED WITH A ". "
7542	033212	033574			OPN. X1		;TERMINATED WITH A ". . "
7543	033214	010264	001506	9\$:	MOV	R2, RPT(R4)	;SAVE THIS VARIABLE
7544	033220	000705			BR	2\$	;MOVE TO NEXT VARIABLE
7545	033222	104401	044042	OPNPAT:	TYPE	, MSG. SP	;TYPE SPACES
7546	033226	104401	043044		TYPE	, MSG. PAT	;TYPE "PAT"
7547	033232	104401	043050		TYPE	, MSG. EQ	;TYPE "="
7548	033236	016446	001506		MOV	RPT(R4), -(SP)	;SAVE RPT(R4) FOR TYPEOUT
7549	033242	104402			TYPOC		;GO TYPE--OCTAL ASCII(ALL DIGITS)
7550	033244	104401	044043		TYPE	, MSG. SP+1	;TYPE ONE SPACE
7551	033250	104411			RDLIN		;READ ASCII STRING
7552	033252	012601			MOV	(SP)+, R1	;PICKUP POINTER
7553	033254	004037	033770		JSR	RO, @#CK. CHR	;CHECK ONE CHARACTER
7554	033260	033222			OPNPAT		;ILLEGAL CHARACTER
7555	033262	033020			OPNPRM		;CARRIAGE RETURN
7556	033264	033316			3\$		;"/"
7557	033266	033020			OPNPRM		;": "
7558	033270	033274			1\$		;". "
7559	033272	033314			2\$		;DIGIT 0-9
7560	033274	105711		1\$:	TSTB	(R1)	; "CR" AFTER THE PERIOD?
7561	033276	001531			BEQ	OPN. N2	;YES--GO CLOSE THIS TEST
7562	033300	122721	000056		CMPB	#'. , (R1)+	;NO--PERIOD?
7563	033304	001346			BNE	OPNPAT	;NO--LOOP
7564	033306	105711			TSTB	(R1)	; "CR" AFTER A DOUBLE PERIOD?
7565	033310	001533			BEQ	OPN. X2	;YES--GO START TESTING
7566	033312	000743			BR	OPNPAT	;NO--LOOP
7567	033314	005301		2\$:	DEC	R1	;BACKUP THE ASCII POINTER
7568	033316			3\$:			
7569	033316	004037	034230		JSR	RO, @#CK. NUM	;CHECK THE NUMBER
7570	033322	033222			OPNPAT		;ILLEGAL INPUT
7571	033324	033332			4\$		;TERMINATED WITH A ", " OR "CR"



```

7572 033326 033556          OPN. N1          ; TERMINATED WITH A ". "
7573 033330 033574          OPN. X1          ; TERMINATED WITH A ". ."
7574 033332 010264 001506 45:  MOV      R2,RPT(R4) ; SAVE THE INPUT NUMBER
7575 033336 006002          ROR      R2      ; OPEN PATTERN 0?
7576 033340 103227          BCC     OPNPRM   ; NO--START AT BEGINNING OF PARAMETER TABLE
7577 033342 104412          SAVREG          ; SAVE R0 - R5
7578 033344 005000          CLP      R0     ; START WITH WORD 0
7579 033346 012704 003104  OPNWDS: MOV     #PATO,R4
7580 033352 15:
7581 033352 104401 033360  TYPE     ,655    ; TYPE ASCIZ STRING
7582 033356 000403          BR      645     ; GET OVER THE ASCIZ
7583
7584 033366 645: .ASCIZ / WD/
7585 033366 010046          MOV     RO,-(SP) ; PUT RO ON THE STACK
7586 033370 004737 023222  JSR     PC,@#SSB2D ; CHANGE RO TO DECIMAL ASCIZ
7587 033374 004737 023452  JSR     PC,@#SSUPRS ; TYPE WITHOUT LEADING ZEROS
7588 033400 104401 043050  TYPE     ,MSG.EQ ; TYPE "="
7589 033404 011446          MOV     (R4),-(SP) ; SAVE (R4) FOR TYPEOUT
7590 033406 104402          TYPOC          ; GO TYPE--OCTAL ASCII(ALL DIGITS)
7591 033410 104411          RDLIN          ; READ ASCIZ STRING
7592 033412 012601          MOV     (SP)+,R1 ; PICKUP THE POINTER
7593 033414 004037 033770  JSR     RO,@#CK.CHR ; CHECK ONE CHARACTER
7594 033420 033352          15           ; ILLEGAL CHARACTER
7595 033422 033454          45           ; CARRIAGE RETURN
7596 033424 033436          25           ; "/"
7597 033426 033454          45           ; "."
7598 033430 033470          55           ; " "
7599 033432 033434          105          ; DIGIT 0-9
7600 033434 005301 105:  DEC     R1      ; BACKUP THE ASCII POINTER
7601 033436 25:
7602 033436 004037 034230  JSR     RO,@#CK.NUM ; CHECK THE NUMBER
7603 033442 033352          15           ; ILLEGAL INPUT
7604 033444 033452          35           ; TERMINATED WITH A ".," OR "CR"
7605 033446 033510          65           ; TERMINATED WITH A ". ."
7606 033450 033522          85           ; TERMINATED WITH A ". ."
7607 033452 010214 35:  MOV     R2,(R4) ; SAVE THE INPUT
7608 033454 005724 45:  TST     (R4)+   ; MOVE TO NEXT WORD
7609 033456 005200          INC     RO     ; INCREMENT THE COUNT
7610 033460 022700 000020  CMP     #16.,RO ; COUNT TO LARGE?
7611 033464 003332          BGT     15     ; NO--BRANCH
7612 033466 000726          BR     OPNWDS  ; YES--BRANCH
7613 033470 105711 55:  TSTB   (R1)    ; "CR" AFTER THE PERIOD?
7614 033472 001407          BEQ     75     ; YES--GO CLOSE THIS TEST
7615 033474 122721 000056  CMPB   #'.,(R1)+ ; NO--PERIOD?
7616 033500 001324          BNE     15     ; NO--BRANCH ILLEGAL INPUT STRING
7617 033502 105711          TSTB   (R1)    ; "CR" AFTER THE "PERIOD-PERIOD"?
7618 033504 001407          BEQ     95     ; YES--GO START TESTING
7619 033506 000721          BR     15     ; NO--LOOP
7620 033510 010224 65:  MOV     R2,(R4)+ ; SAVE THE INPUT
7621 033512 004737 033534 75:  JSR     PC,@#CLSWDS ; CLOSE THE DATA PATTERN
7622 033516 104413          RESREG          ; RESTORE R0 - R5
7623 033520 000420          BR     OPN.N2  ; MOVE TO NEXT TEST
7624 033522 010224 85:  MOV     R2,(R4)+ ; SAVE THE INPUT
7625 033524 004737 033534 95:  JSR     PC,@#CLSWDS ; CLOSE THE DATA PATTERN
7626 033530 104413          RESREG          ; RESTORE R0 - R5
7627 033532 000422          BR     OPN.X2  ; START TESTING
    
```

```

7628 033534 012701 003104 CLSWDS: MOV #PATO,R1 ;FIRST ADDRESS OF DATA PATTERN
7629 033540 005200 15: INC R0 ;COUNT THE LAST WORD THAT WAS STORED
7630 033542 022700 000017 CMP #15,R0 ;END OF TABLE
7631 033546 002402 BLT 25 ;YES--EXIT
7632 033550 012124 MOV (R1)+,(R4)+ ;COPY
7633 033552 000772 BR 15 ;LOOP
7634 033554 000207 25: RTS PC ;RETURN
7635 033556 010264 001506 OPN. N1: MOV R2,RPT(R4) ;SAVE THIS VARIABLE
7636 033562 005726 OPN. N2: TST (SP)+ ;CLEAN OFF THE STACK
7637 033564 004737 033634 JSR PC,CLOSE ;CLOSE THIS TEST
7638 033570 000137 032656 JMP OPN.1 ;GO OPEN THE NEXT TEST
7639 033574 010264 001506 OPN. X1: MOV R2,RPT(R4) ;SAVE THIS VARIABLE
7640 033600 005726 OPN. X2: TST (SP)+ ;CLEAN OFF THE STACK
7641 033602 004737 033634 15: JSR PC,CLOSE ;CLOSE THIS TEST
7642 033606 005725 25: TST (R5)+ ;UPDATE THE INDEX
7643 033610 020527 000034 CMP R5,#16*2 ;INDEX TO BIG?
7644 033614 002403 BLT 35 ;NO--BRANCH
7645 033616 104413 RESREG ;RESTORE R0 - R5
7646 033620 000137 032010 JMP GT.PR2 ;GO TO EXIT
7647 033624 036503 001424 35: BIT BITS(R5),R3 ;IS THIS TEST OPEN FOR CHANGE?
7648 033630 001364 BNE 15 ;YES--GO CLOSE IT
7649 033632 000765 BR 25 ;NO--MOVE TO NEXT TEST
7650 033634 104412 CLOSE: SAVREG ;SAVE R0 - R5
7651 033636 012700 001504 MOV #PRM,R0 ;"FROM" ADDRESS
7652 033642 016501 001536 MOV PRMPT(R5),R1 ;"TO" ADDRESS
7653 033646 012002 MOV (R0)+,R2 ;"FROM" INDICATOR
7654 033650 012103 MOV (R1)+,R3 ;"TO" INDICATOR
7655 033652 012704 000001 MOV #1,R4 ;TEST BIT START A "RPT"
7656 033656 030402 15: BIT R4,R2 ;PARAMETER TO BE MOVED?
7657 033660 001403 BEQ 25 ;NO--BRANCH
7658 033662 030403 BIT R4,R3 ;A PLACE TO PUT IT?
7659 033664 001404 BEQ 35 ;NO--BRANCH
7660 033666 011011 MOV (R0),(R1) ;YES--MOVE "FROM" TO "TO"
7661 033670 030403 25: BIT R4,R3 ;"TO" PARAMETER?
7662 033672 001401 BEQ 35 ;NO--BRANCH
7663 033674 005721 TST (R1)+ ;YES--UPDATE THE POINTER
7664 033676 005720 35: TST (R0)+ ;UPDATE FROM POINTER
7665 033700 006304 ASL R4 ;POSITION THE TEST BIT
7666 033702 032704 002000 BIT #BIT10,R4 ;DONE?
7667 033706 001763 BEQ 15 ;NO--BRANCH
7668 033710 104413 RESREG ;RESTORE R0 - R5
7669 033712 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 033714 121127 000060 CK. OCT: CMPB (R1),#'0 ;LESS THAN ZERO?
7680 033720 103407 BLO 15 ;YES -- BRANCH
7681 033722 121127 000067 CMPB (R1),#'7 ;GREATER THAN SEVEN?
7682 033726 101004 BHI 15 ;YES -- BRANCH
7683 033730 111102 MOVB (R1),R2 ;GET THE CHARACTER
  
```



```

7684 033732 042702 177770          BIC      # C7,R2          ;STRIP AWAY THE ASCII
7685 033736 005720                    TST      (R0)+          ;ADJUST FOR RETURN
7686 033740 000200          15:      RTS      RO          ;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          ;
7692          ;      MOV      #ADR,R1          ;ADDRESS OF ASCII CHARACTER
7693          ;      JSR      RO, @#CK. DEC ;CHECK THE CHARACTER
7694          ;      RETURN1 ;NOT BETWEEN 0 AND 9
7695          ;      RETURN2 ;BETWEEN 0 AND 9
7696          ;      ;R2 = DIGIT
7697 033742 121127 000060          CK. DEC: CMPB     (R1),# '0          ;LESS THAN ZERO?
7698 033746 103407                    BLO      15             ;YES -- BRANCH
7699 033750 121127 000071          CMPB     (R1),# '9          ;GREATER THAN NINE?
7700 033754 101004                    BHI      15             ;YES -- BRANCH
7701 033756 111102                    MOVB     (R1),R2          ;GET THE CHARACTER
7702 033760 042702 000060          BIC      # '0,R2          ;STRIP AWAY THE ASCII
7703 033764 005720                    TST      (R0)+          ;ADJUST FOR RETURN
7704 033766 000200          15:      RTS      RO          ;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      RO, @#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 033770 105711          CK. CHR: TSTB     (R1)          ;"CARRIAGE RETURN"?
7720 033772 001420                    BEQ      45             ;YES -- BRANCH
7721 033774 121127 000057          CMPB     (R1),# '/'          ;"SLASH"?
7722 034000 001414                    BEQ      35             ;YES -- BRANCH
7723 034002 121127 000054          CMPB     (R1),# ','          ;"COMMA"?
7724 034006 001410                    BEQ      25             ;YES -- BRANCH
7725 034010 121127 000056          CMPB     (R1),# '.'          ;"PERIOD"?
7726 034014 001404                    BEQ      15             ;YES -- BRANCH
7727 034016 004037 033742          JSR      RO, @#CK. DEC          ;"DIGIT"?
7728 034022 000406                    BR       55             ;NO -- BRANCH
7729 034024 005720                    TST      (R0)+          ;DIGIT BETWEEN 0-9
7730 034026 005720          15:      TST      (R0)+          ;PERIOD
7731 034030 005720          25:      TST      (R0)+          ;COMMA
7732 034032 005720          35:      TST      (R0)+          ;SLASH
7733 034034 005720          45:      TST      (R0)+          ;CARRIAGE RETURN
7734 034036 005201                    INC      R1             ;MOVE POINTER TO NEXT CHARACTER
7735 034040 011000          55:      MOV      (R0),RO          ;UNKNOWN CHARACTER
7736 034042 000200          RTS      RO          ;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 ;
7742 ;
7743 ;
7744 ;
7745 ;
7746 ;
7747 ;
7748 ;
7749 ;
7750 034044 010446 CK. DIG: MOV R4,-(SP) ;SAVE R4
7751 034046 010346 MOV R3,-(SP) ;SAVE R3
7752 034050 010246 MOV R2,-(SP) ;SAVE THE MAX. SIZE ON THE STACK
7753 034052 005002 CLR R2 ;START WITH 0
7754 034054 005003 CLR R3
7755 034056 005004 CLR R4
7756 034060 004037 033770 JSR RO,@#CK. CHR ;CHECK ONE CHARACTER
7757 034064 034214 85 ;ILLEGAL CHARACTER
7758 034066 034214 85 ;CARRIAGE RETURN
7759 034070 034214 85 ;"/"
7760 034072 034214 85 ;" "
7761 034074 034214 85 ;" "
7762 034076 034100 15 ;DIGIT 0-9
7763 034100 006303 15: ASL R3 ;2
7764 034102 010346 MOV R3,-(SP) ;SAVE #2
7765 034104 006303 ASL R3 ;4
7766 034106 006303 ASL R3 ;8
7767 034110 062603 ADD (SP)+,R3 ;(#8)+(#2)=#10.
7768 034112 060203 ADD R2,R3 ;UPDATE THE INPUT NUMBER
7769 034114 004037 033770 JSR RO,@#CK. CHR ;CHECK ONE CHARACTER
7770 034120 034214 85 ;ILLEGAL CHARACTER
7771 034122 034134 95 ;CARRIAGE RETURN
7772 034124 034214 85 ;"/"
7773 034126 034142 35 ;" "
7774 034130 034140 25 ;" "
7775 034132 034100 15 ;DIGIT 0-9
7776 034134 005301 95: DEC R1 ;BACKUP THE CHARACTER POINTER
7777 034136 000401 BR 35 ;CONTINUE
7778 034140 005724 25: TST (R4)+ ;"PERIOD"
7779 034142 005724 35: TST (R4)+ ;"COMMA" OR "CR"
7780 034144 004037 033770 JSR RO,@#CK. CHR ;CHECK ONE CHARACTER
7781 034150 034214 85 ;ILLEGAL CHARACTER
7782 034152 034204 65 ;CARRIAGE RETURN
7783 034154 034214 85 ;"/"
7784 034156 034214 85 ;" "
7785 034160 034164 45 ;" "
7786 034162 034174 55 ;DIGIT 0-9
7787 034164 005724 45: TST (R4)+ ;"PERIOD-PERIOD"
7788 034166 105711 TSTB (R1) ;"CR"?
7789 034170 001405 BEQ 65 ;YES--BRANCH
7790 034172 000410 BR 85
7791 034174 126127 177776 000054 55: CMPB -2(R1),#' ;WAS CHARACTER BEFORE THE DIGIT A COMMA?
7792 034202 001004 BNE 85 ;NO--EXIT
7793 034204 020316 65: CMP R3,(SP) ;INPUT TO LARGE?
7794 034206 101001 BHI 75 ;YES -- BRANCH
7795 034210 060400 ADD R4,RO ;ADJUST RETURN ADDRESS
  
```



```

7796 034212 005720          75:   TST      (R0)+
7797 034214 010302          85:   MOV      R3,R2          ;NUMBER TO R2
7798 034216 005726          TST      (SP)+          ;CLEAN MAX. SIZE OFF OF STACK
7799 034220 012603          MOV      (SP)+,R3        ;RESTORE R3
7800 034222 012604          MOV      (SP)+,R4        ;RESTORE R4
7801 034224 011000          MOV      (R0),R0         ;GET RETURN ADDRESS
7802 034226 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 034230 010346          CK. NUM: MOV      R3,-(SP)    ; SAVE R3
7815 034232 005003          CLR      R3              ; START NUMBER AT ZERO
7816 034234 004037 033714  JSR      R0,@#CK.OCT     ; OCTAL DIGIT?
7817 034240 000440          BR      65               ; NO--BRANCH
7818 034242 005201          15:   INC      R1              ; MOVE TO NEXT CHARACTER
7819 034244 006303          ASL      R3              ; FOR THE OCTAL NUMBER IN R3
7820 034246 103435          BCS      65               ; DON'T LET IT GET TO BIG
7821 034250 006303          ASL      R3
7822 034252 103433          BCS      65
7823 034254 006303          ASL      R3
7824 034256 103431          BCS      65
7825 034260 060203          ADD      R2,R3
7826 034262 004037 033714  JSR      R0,@#CK.OCT     ; IS THIS AN OCTAL DIGIT?
7827 034266 000401          BR      25               ; NO--FIND OUT WHAT IT IS
7828 034270 000764          BR      15               ; YES--MAKE IT PART OF THE NUMBER
7829 034272 010302          25:   MOV      R3,R2          ; SAVE THE OCTAL NUMBER
7830 034274 005003          CLR      R3              ; START WITH ZERO INDEX
7831 034276 004037 033770  JSR      R0,@#CK.CHR     ; CHECK ONE CHARACTER
7832 034302 034342          65          ; ILLEGAL CHARACTER
7833 034304 034332          55          ; CARRIAGE RETURN
7834 034306 034342          65          ; "/"
7835 034310 034332          55          ; ","
7836 034312 034316          35          ; "."
7837 034314 034342          65          ; DIGIT 0-9
7838 034316 005723          35:   TST      (R3)+          ; "PERIOD"
7839 034320 121127 000056  CMPB    (R1),#'.         ; "PERIOD-PERIOD"?
7840 034324 001002          BNE      55               ; NO--BRANCH
7841 034326 005201          INC      R1              ; YES--ADVANCE THE POINTER
7842 034330 005723          45:   TST      (R3)+          ; "PERIOD-PERIOD"
7843 034332 005723          55:   TST      (R3)+          ; "COMMA"
7844 034334 105711          TSTB    (R1)             ; "CR"?
7845 034336 001001          BNE      65               ; NO--BRANCH
7846 034340 060300          ADD      R3,R0           ; YES--SAVE THE OCTAL NUMBER
7847 034342 012603          65:   MOV      (SP)+,R3        ; RESTORE R3
7848 034344 011000          MOV      (R0),R0         ; PICKUP EXIT ADDRESS
7849 034346 000200          RTS      R0              ; RETURN

```

```
7850 ; ; *****  
7851  
7852  
7853 .SBTTL SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)  
7854  
7855 ; COPYRIGHT (C) 1976  
7856 ; DIGITAL EQUIPMENT CORP.  
7857 ; MAYNARD, MA 01754  
7858 ; AUTHOR(S): JIM LACEY/CHUCK HESS  
7859  
7860 ; ; *****  
7861  
7862 ; STORAGE FOR RPDS1, RPER1, RPER2, AND RPER3 ON AN ERROR "2"  
7863 ; RPERRS = RPDS1  
7864 ; RPERRS+2 = RPER1  
7865 ; RPERRS+4 = RPER2  
7866 ; RPERRS+6 = RPER3  
7867  
7868 034350 000000 000000 000000 RPERRS: .WORD 0,0,0,0  
7869 034356 000000  
7870  
7871 ; TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)  
7872 ; DRVACT=0 IF DRIVE IS IDLE  
7873 ; DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND  
7874 ; DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION  
7875  
7876 034360 000 DRVACT: .BYTE 0 ; DRIVE 0  
7877 034361 000 .BYTE 0 ; DRIVE 1  
7878 034362 000 .BYTE 0 ; DRIVE 2  
7879 034363 000 .BYTE 0 ; DRIVE 3  
7880 034364 000 .BYTE 0 ; DRIVE 4  
7881 034365 000 .BYTE 0 ; DRIVE 5  
7882 034366 000 .BYTE 0 ; DRIVE 6  
7883 034367 000 .BYTE 0 ; DRIVE 7  
7884  
7885 ; TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)  
7886 ; DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT  
7887 ; DRVSTA>0 IF DRIVE IS ONLINE  
7888 ; DRVSTA<0 IF DRIVE IS UNSAFE  
7889  
7890 034370 000 DRVSTA: .BYTE 0 ; DRIVE 0  
7891 034371 000 .BYTE 0 ; DRIVE 1  
7892 034372 000 .BYTE 0 ; DRIVE 2  
7893 034373 000 .BYTE 0 ; DRIVE 3  
7894 034374 000 .BYTE 0 ; DRIVE 4  
7895 034375 000 .BYTE 0 ; DRIVE 5  
7896 034376 000 .BYTE 0 ; DRIVE 6  
7897 034377 000 .BYTE 0 ; DRIVE 7  
7898  
7899 ; TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)  
7900 ; DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)  
7901 ; DRVTYP=1 IF DRIVE IS RPO4  
7902 ; DRVTYP=2 IF DRIVE IS RPO5  
7903 ; DRVTYP=4 IF DRIVE IS RPO6  
7904 ; DRVTYP=-1 IF NOT RPO4/5/6  
7905
```



```
7906 034400 000 DRVTYP: . BYTE 0 ;DRIVE 0
7907 034401 000 . BYTE 0 ;DRIVE 1
7908 034402 000 . BYTE 0 ;DRIVE 2
7909 034403 000 . BYTE 0 ;DRIVE 3
7910 034404 000 . BYTE 0 ;DRIVE 4
7911 034405 000 . BYTE 0 ;DRIVE 5
7912 034406 000 . BYTE 0 ;DRIVE 6
7913 034407 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 034410 000 DPINT: . BYTE 0 ;DRIVE 0
7920 034411 000 . BYTE 0 ;DRIVE 1
7921 034412 000 . BYTE 0 ;DRIVE 2
7922 034413 000 . BYTE 0 ;DRIVE 3
7923 034414 000 . BYTE 0 ;DRIVE 4
7924 034415 000 . BYTE 0 ;DRIVE 5
7925 034416 000 . BYTE 0 ;DRIVE 6
7926 034417 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 034420 000 DPRQS: . BYTE 0 ;DRIVE 0
7933 034421 000 . BYTE 0 ;DRIVE 1
7934 034422 000 . BYTE 0 ;DRIVE 2
7935 034423 000 . BYTE 0 ;DRIVE 3
7936 034424 000 . BYTE 0 ;DRIVE 4
7937 034425 000 . BYTE 0 ;DRIVE 5
7938 034426 000 . BYTE 0 ;DRIVE 6
7939 034427 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 034430 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 034432 000000 SRCHWT: . WORD 0
7954
7955 ;RPO4/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
7956 ;ACTDRV=0 IF DRIVER IS INACTIVE
7957 ;ACTDRV>0 IF DRIVER IS ACTIVE
7958
7959 034434 000 ACTDRV: . BYTE 0
7960
7961 ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
```

```

7962                                     ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
7963                                     ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
7964
7965 034435      000      ACTSTR: .BYTE  0
7966
7967                                     ;UNLOAD FLAG (ULDFLG=8 BYTES)
7968                                     ;ULDFLG=0 IF NO UNLOAD COMMAND
7969                                     ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
7970                                     ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
7971
7972 034436      000      ULDFLG: .BYTE  0          ;DRIVE 0
7973 034437      000      .BYTE  0          ;DRIVE 1
7974 034440      000      .BYTE  0          ;DRIVE 2
7975 034441      000      .BYTE  0          ;DRIVE 3
7976 034442      000      .BYTE  0          ;DRIVE 4
7977 034443      000      .BYTE  0          ;DRIVE 5
7978 034444      000      .BYTE  0          ;DRIVE 6
7979 034445      000      .BYTE  0          ;DRIVE 7
7980
7981                                     ;LOOK AHEAD COUNT (LACNT=8 BYTES)
7982                                     ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
7983
7984 034446      000      LACNT:  .BYTE  0          ;DRIVE 0
7985 034447      000      .BYTE  0          ;DRIVE 1
7986 034450      000      .BYTE  0          ;DRIVE 2
7987 034451      000      .BYTE  0          ;DRIVE 3
7988 034452      000      .BYTE  0          ;DRIVE 4
7989 034453      000      .BYTE  0          ;DRIVE 5
7990 034454      000      .BYTE  0          ;DRIVE 6
7991 034455      000      .BYTE  0          ;DRIVE 7
7992
7993                                     ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
7994                                     ;SAVEFG <0 IF SAVE THE RH11/RPO4/5/6 REGISTERS WHEN THE
7995                                     ;OPERATION IS COMPLETED AS PER (DPB+14).
7996                                     ;SAVEFG=0 IF SAVE THE RH11/RPO4/5/6 REGISTERS, AS PER
7997                                     ;(DPB+14), AFTER AN ERROR.
7998
7999 034456      000000  SAVEFG: .WORD  0
8000
8001                                     ;SEEK FLAG (SEEKFG=1 WORD)
8002                                     ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
8003                                     ;FOR A DATA TRANSFER START A SEARCH COMMAND
8004                                     ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
8005                                     ;DISREGARD THE WINDOW
8006
8007 034460      000000  SEEKFG: .WORD  0
8008
8009                                     ;TIMEOUT TABLE (TIMER=8 WORDS)
8010                                     ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
8011
8012 034462      177777  TIMER:  .WORD -1          ;DRIVE 0
8013 034464      177777  .WORD -1          ;DRIVE 1
8014 034466      177777  .WORD -1          ;DRIVE 2
8015 034470      177777  .WORD -1          ;DRIVE 3
8016 034472      177777  .WORD -1          ;DRIVE 4
8017 034474      177777  .WORD -1          ;DRIVE 5

```



```

8018 034476 177777          .WORD -1      ;DRIVE 6
8019 034500 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 034502 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 034504 001          ATABIT: .BYTE 1      ;DRIVE 0
8032 034505 002          .BYTE 2      ;DRIVE 1
8033 034506 004          .BYTE 4      ;DRIVE 2
8034 034507 010          .BYTE 10     ;DRIVE 3
8035 034510 020          .BYTE 20     ;DRIVE 4
8036 034511 040          .BYTE 40     ;DRIVE 5
8037 034512 100          .BYTE 100    ;DRIVE 6
8038 034513 200          .BYTE 200    ;DRIVE 7
8039
8040          ;RPO4/5/6 TO RH11 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
8041          ;CALLING IT FATAL (MCPEMX=1 WORD)
8042
8043 034514 000003          MCPEMX: .WORD 3
8044
8045          ;STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RPO4/5/6),
8046          ;RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).
8047
8048 034516 176700          RPADR: .WORD 176700
8049 034520 000254 000240  RPVEC: .WORD 254,5*32
8050
8051          ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
8052
8053 034524 000004          MXLACT: .WORD 4
8054          ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
8055
8056 034526 001000          MXDLTA: .WORD 8.*64.
8057          ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
8058
8059 034530 000200          MNDLTA: .WORD 2*64.
8060          ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
8061
8062 034532 000005          MXWNDW: .WORD 5
8063
8064          ;DEFINITIONS OF THE RH11/RPO4/5/6 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/RPO4/5/6 DRIVER INITIALIZATION CODE		
8088			;THIS ROUTINE WILL DETERMINE WHICH RPO4/5/6 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	034534	104412	RPINIT: SAVREG		;SAVE R0 - R5
8101	034536	013746	MOV @#PS,-(SP)		;SAVE THE PRESENT PROCESSOR STATUS
8102	034542	012737	MOV #<5*32.>,@#PS	177776	;CHANGE THE PRIORITY TO 5
8103	034550	004737	JSR PC,CLRQUE	042520	;CLEAR ALL REQUEST QUEUES
8104	034554	012701	MOV #RPERRS,R1	034350	;FIRST ADDRESS TO BE CLEARED
8105	034560	012702	MOV #SEEKFG,R2	034460	;LAST ADDRESS TO BE CLEARED
8106	034564	005021	15: CLR (R1)+		;CLEAR
8107	034566	020102	15: CMP R1,R2		;ARE WE DONE?
8108	034570	101775	BLOS 15		;BRANCH IF NO
8109	034572	012702	MOV #DTUW,R2	034502	;LAST ADDRESS
8110	034576	012721	25: MOV #-1,(R1)+	177777	;INITIALIZE
8111	034602	020102	25: CMP R1,R2		;DONE?
8112	034604	101774	BLOS 25		;LOOP IF NO
8113	034606	005037	CLR DRVSTA	034370	;SET ALL DRIVES TO OFFLINE
8114	034612	005037	CLR DRVSTA+2	034372	
8115	034616	005037	CLR DRVSTA+4	034374	
8116	034622	005037	CLR DRVSTA+6	034376	
8117	034626	013703	MOV RPVEC,R3	034520	;SETUP THE RH11/RPO4/5/6 VECTOR
8118	034632	012723	MOV #ISR,(R3)+	037400	
8119	034636	013713	MOV RPVEC+2,(R3)	034522	
8120	034642	013704	MOV RPADR,R4	034516	;FIRST ADDRESS OF RH11/RPO4
8121	034646	012764	MOV #BIT05,RPCS2(R4)	000040	;MASSBUS INIT
8122	034654	005001	CLR R1	000010	;START WITH DRIVE 0
8123	034656	004037	35: JSR R0,DRVINT	034746	;INIT THE DRIVE
8124	034662	000401	BR 45		; 'DVA' NOT SET OR PARITY ERROR
8125	034664	000402	BR 55		;NORMAL RETURN
8126	034666	105061	45: CLRB DRVSTA(R1)	034370	;SET DRIVE STATUS TO OFFLINE
8127	034672	005201	55: INC R1		;GO TO NEXT DRIVE
8128	034674	042701	BIC #C7,R1	177770	;MASK OUT UNUSED BITS
8129	034700	001366	BNE 35		;BR IF MORE DRIVES TO GO



```

8130 034702 012701 000007      MOV      #7,R1          ;START WITH DRIVE 7
8131 034706 005037 177776      CLR      @#PS          ;CLEAR THE PROCESSOR STATUS
8132 034712 105761 034410      65:     TSTB     DPINT(R1) ;WAITING FOR DRIVE TO SWITCH PORTS ?
8133 034716 001405                BEQ      85            ;BR NOT WAITING
8134 034720 004737 042154      JSR      PC,SET. IE    ;SET INTERRUPT
8135 034724 105761 034410      75:     TSTB     DPINT(R1) ;DRIVE SWITCHED PORTS ?
8136 034730 001375                BNE     75            ;BR IF NOT
8137 034732 005301                DEC     R1            ;GO TO THE NEXT DRIVE
8138 034734 100366                BPL     65            ;CHECK NEXT DRIVE
8139 034736 012637 177776      MOV      (SP)+,@#PS    ;RESTORE THE PROCESSOR STATUS
8140 034742 104413                RESREG                ;RESTORE R0 - R5
8141 034744 000207                RTS     PC            ;BYE-BYE
8142
8143 ;DRIVE INITIALIZATION ROUTINE
8144 ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
8145 ;AN RPO4/5/6. 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
8150 ;CALL
8151 ;
8152 ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
8153 ;      MOV      RPADR,R4        ;UNIBUS ADDRESS OF RH11/RPO4/5/6 (RPCS1)
8154 ;      JSR      RO,DRVINT       ;CALLED BY A JSR
8155 ;      RETURN1      ;ERROR OCCURRED (PARITY)
8156 ;      RETURN2      ;NORMAL RETURN
8157
8158 ;DRVINT: MOV      R5,-(SP)      ;SAVE R5
8159 ;      CLRB     DRVSTA(R1)     ;START DRIVE STATUS AS OFFLINE
8160 ;      CLRB     DRVSTYP(R1)    ;CLEAR THE DRIVE TYPE INDICATOR
8161 ;      CLRB     ULDFLG(R1)     ;CLEAR THE UNLOAD FLAG
8162 ;      MOV      R1,RPCS2(R4)    ;SELECT A DRIVE
8163 ;      MOVB    #111,RPCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
8164 ;      BIT     #BIT12,RPCS2(R4) ;NONEXISTENT DRIVE?
8165 ;      BEQ     15              ;NO---BRANCH
8166 ;      JSR     PC,SET. IE      ;GO SET "IE" WITHOUT A "TRE"
8167 ;      BR     65              ;LEAVE THIS ROUTINE
8168 ;      15:    CLRB     DRVSTA(R1) ;SET DRIVE STATUS TO OFFLINE
8169 ;      BIT     #BIT11,RPCS1(R4) ;SEE IF DRIVE AVAILABLE
8170 ;      BEQ     75              ;BR IF DRIVE NOT AVAILABLE
8171 ;      JSR     RO,RD. RP       ;READ THE DRIVE TYPE REG.
8172 ;      RPD     85              ;ERROR RETURN ADDRESS
8173 ;      MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
8174 ;      MOVB   #1,DRVSTYP(R1)  ;SET RPO4 INDICATOR
8175 ;      CMP     #20020,R5       ;IS IT A SINGLE PORT RPO4?
8176 ;      BEQ     25              ;BRANCH IF YES
8177 ;      CMP     #24020,R5       ;IS IT A DUAL PORT RPO4?
8178 ;      BEQ     25              ;BR IF YES
8179 ;      MOVB   #2,DRVSTYP(R1)  ;SET RPO5 INDICATOR
8180 ;      CMP     #20021,R5       ;SINGLE PORT RPO5 ?
8181 ;      BEQ     25              ;BR IF YES
8182 ;      CMP     #24021,R5       ;DUAL PORT RPO5 ?
8183 ;      BEQ     25              ;BR IF YES
8184 ;      MOVB   #4,DRVSTYP(R1)  ;SET RPO6 INDICATOR
8185 ;      CMP     #20022,R5       ;SINGLE PORT RPO6 ?
    
```

```

8186 035120 001407      BEQ      25          ;BR IF YES
8187 035122 022705 024022  CMP      #24022,R5   ;DUAL PORT RPO6 ?
8188 035126 001404      BEQ      25          ;BR IF YES
8189 035130 112761 177777 034400  MOVB    #-1,DRVTP(R1) ;SET INDICATOR TO 'OTHER'
8190 035136 000446      BR       65          ;EXIT
8191 035140 012746 000121      25:    MOV     #121,-(SP)   ;DO A "READ-IN PRESET"
8192 035144 004037 041644      JSR     RO,WRT.RP
8193 035150 000000      RPCS1
8194 035152 035300      85
8195 035154 012746 010000      MOV     #BIT12,-(SP) ;SET FMT22=1
8196 035160 004037 041644      JSR     RO,WRT.RP
8197 035164 000032      RPOF
8198 035166 035300      85
8199 035170 004037 041464      JSR     RO,RO.RP    ;READ RPDS1
8200 035174 000012      RPDS1
8201 035176 035300      85
8202 035200 012605      MOV     (SP)+,R5     ;AND SAVE IT IN R5
8203 035202 100015      BPL     45          ;BRANCH IF ATA=0
8204 035204 116164 034504 000016  MOVB    ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
8205 035212 004037 041464      JSR     RO,RO.RP    ;FIND OUT WHY ATA=1
8206 035216 000014      RPER1
8207 035220 035300      85
8208 035222 006126      ROL     (SP)+       ;IS IT UNSAFE?
8209 035224 100004      BPL     45          ;BR IF NOT
8210 035226 112761 177777 034370  MOVB    #-1,DRVSTA(R1) ;SET UNSAFE INDICATOR
8211 035234 000407      BR       65          ;EXIT
8212 035236 005105      45:    COM     R5          ;CHECK MOL, DPR, DRY, AND VV
8213 035240 042705 167077      BIC     #(<BIT12!BIT08!BIT07!BIT06>),R5
8214 035244 001003      BNE     65          ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
8215 035246 112761 000001 034370  MOVB    #1,DRVSTA(R1) ;SET DRIVE STATUS TO ONLINE
8216 035254 005720      65:    TST     (RO)+       ;STEP OVER THE ERROR RETURN
8217 035256 000410      BR       85          ;EXIT
8218 035260 006301      75:    ASL     R1          ;CHANGE INDEX TO ADDRESS WORDS
8219 035262 012761 003720 034462  MOV     #2000.,TIMER(R1) ;START 2 SEC TIMER
8220 035270 006201      ASR     R1          ;RESTORE R1
8221 035272 105161 034410      COMB    DPINT(R1)
8222 035276 005720      TST     (RO)+
8223 035300 012605      85:    MOV     (SP)+,R5     ;RESTORE R5
8224 035302 000200      RTS     RC          ;EXIT
8225
8226      ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
8227      ;
8228      ;CALL
8229      ;
8230      ;      JSR     RO,@#RPO4    ;CALL THE RPO4/5/6 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 035304 013746 177776      RPO4:  MOV     @#PS,-(SP)   ;SAVE THE CALLING STATUS
8237 035310 013737 034522 177776  MOV     RPVEC+2,@#PS ;DON'T ALLOW ANY RPO4/5/6 INTERRUPTS
8238 035316 112737 000001 034434  MOVB    #1,ACTDRV   ;SET "ACTIVE DRIVER" FLAG
8239 035324 104412      SAVREG ;SAVE RO - R5
8240 035326 011002      MOV     (RO),R2     ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
8241 035330 005062 000016      CLR     16(R2)     ;CLEAR THE STATUS/ERROR INDICATOR
    
```



```

8242 035334 111201          MOV      (R2),R1          ;PICKUP THE DRIVE NUMBER
8243 035336 013704 034516    MOV      RPADR,R4        ;UNIBUS ADDRESS OF RPCS1
8244 035342 105761 034370    TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
8245 035346 003014          BGT      15              ;BRANCH IF ONLINE
8246 035350 105761 034436    TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
8247 035354 001036          BNE      35              ;BRANCH IF YES
8248 035356 105761 034410    TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
8249 035362 001042          BNE      55              ;BR IF YES
8250 035364 004037 034746    JSR      RO,DRVINT       ;GO INIT. THE DRIVE
8251 035370 000434          BR       45              ;ERROR RETURN
8252 035372 105761 034370    TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
8253 035376 003445          BLE      65              ;BR IF NOT
8254 035400 105761 034420      15:    TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
8255 035404 001031          BNE      55              ;BR IF YES
8256 035406 010164 000010    MOV      R1,RPCS2(R4)    ;SELECT THE DRIVE
8257 035412 004037 042616    JSR      RO,DRVQLE       ;PUT THIS REQUEST IN QUEUE
8258 035416 000460          BR       95              ;QUEUE IS FULL
8259 035420 122762 000103 000002  CMPB    #103,2(R2)       ;IS THIS REQ. FOR AN UNLOAD?
8260 035426 001003          BNE      25              ;BR IF NO
8261 035430 112761 177777 034436    MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
8262 035436 105761 034360      25:    TSTB    DRVACT(R1)       ;IS THIS DRIVE ACTIVE?
8263 035442 001043          BNE      85              ;BR IF YES
8264 035444 004737 035576    JSR      PC,OPT          ;CALL THE OPTIMIZER
8265 035450 000440          BR       85
8266 035452 012762 120000 000016  35:    MOV      #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
8267 035460 000434          BR       85              ;EXIT
8268 035462 004737 036706      45:    JSR      PC,C17          ;GO HANDLE THE PARITY ERROR
8269 035466 000431          BR       85
8270 035470 004037 042616      55:    JSR      RO,DRVQLE       ;PUT REQUEST IN QUEUE
8271 035474 000431          BR       95              ;QUEUE IS FULL
8272 035476 032714 000100    BIT     #BIT06,(R4)      ;IS 'IE' SET ALREADY ?
8273 035502 001023          BNE      85              ;BR IF IT IS
8274 035504 004737 042154    JSR      PC,SET.IE       ;SET INTERRUPT
8275 035510 000420          BR       85              ;RETURN, REQUEST IN QUEUE
8276 035512 105761 034370      65:    TSTB    DRVSTA(R1)       ;SEE IF DRIVE OFFLINE OR UNSAFE
8277 035516 002412          BLT     75              ;BR IF UNSAFE
8278 035520 012762 140000 000016    MOV      #BIT15!BIT14,16(R2) ;SET OFFLINE ERROR INDICATOR
8279 035526 105761 034400    TSTB    DRVSTYP(R1)     ;SEE IF OFFLINE OR NONEXISTENT
8280 035532 001007          BNE      85              ;BR IF OFFLINE
8281 035534 012762 100002 000016    MOV      #BIT15!BIT01,16(R2) ;REPORT DRIVE NONEXISTENT
8282 035542 000403          BR       85              ;GO TO EXIT
8283 035544 012762 110000 000016  75:    MOV      #BIT15!BIT12,16(R2) ;DRIVE IS UNSAFE
8284 035552 104413      85:    RESREG                    ;RESTORE RO - R5
8285 035554 005720          TST     (RO)+            ;SETUP FOR NORMAL RETURN
8286 035556 000401          BR     105              ;FINISH UP, THEN EXIT
8287 035560 104413      95:    RESREG                    ;RESTORE RO - R5
8288 035562 005720      105:   TST     (RO)+            ;CORRECT THE RETURN ADDRESS
8289 035564 105037 034434    CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
8290 035570 012637 177776    MOV     (SP)+, @#PS      ;RETURN "PS" TO USER LEVEL
8291 035574 000200          RTS     RO              ;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 035576 104412 OPT: SAVREG ;SAVE R0 - R5
8300 035600 013746 177776 MOV @#PS, -(SP) ;SAVE PROC. STATUS
8301 035604 146137 034504 034432 BICB ATABIT(R1), SRCHMT ;CLEAR "SEARCH WAIT" KEY
8302 035612 004737 042672 JSR PC, GETREQ ;GET "DPB" POINTER OF REQUEST
8303 035616 005702 TST R2 ;IS THERE A REQUEST IN QUEUE?
8304 035620 001505 BEQ 75 ;NO--BRANCH TO EXIT
8305 035622 032764 004000 000000 BIT #BIT11, RPCS1(R4) ;IS DVA STILL SET ?
8306 035630 001407 BEQ 105 ;BR IF NOT
8307 035632 032764 000100 000012 BIT #BIT6, RPDS1(R4) ;IS VV SET ?
8308 035640 001003 BNE 105 ;BR IF IT IS
8309 035642 004037 034746 95: JSR RO, DRVINT ;SEE IF DRIVE STILL ONLINE ?
8310 035646 000470 BR 65 ;PARITY OR 'DVA' NOT SET
8311 035650 105761 034370 105: TSTB DRVSTA(R1) ;IS DRIVE ONLINE?
8312 035654 003014 BGT 15 ;YES--BRANCH
8313 035656 004737 042714 JSR PC, POPQUE ;NO--REMOVE REQUEST FROM QUEUE
8314 035662 012762 140000 000016 MOV #BIT15!BIT14, 16(R2) ;SET OFFLINE STATUS/ERROR INDICATOR
8315 035670 105761 034370 TSTB DRVSTA(R1) ;IS DRIVE UNSAFE ?
8316 035674 100064 BPL 85 ;BR TO EXIT IF NOT
8317 035676 012762 110000 000016 MOV #BIT15!BIT12, 16(R2) ;SET UNSAFE STATUS/ERROR INDICATOR
8318 035704 000460 BR 85 ;BRANCH TO EXIT
8319 035706 012746 000111 15: MOV #111, -(SP) ;LOAD COMMAND ONTO THE STACK
8320 035712 004037 041644 JSR RO, WRT. RP ;LOAD THE REGISTER
8321 035716 000000 RPCS1 ;REGISTER INCREMENT
8322 035720 036030 65 ;ERROR RETURN ADDRESS
8323 035722 032714 004000 BIT #BIT11, (R4) ;DRIVE AVAILABLE ?
8324 035726 001427 BEQ 55 ;BR IF NOT
8325 035730 122762 000150 000002 CMPB #150, 2(R2) ;IS THE REQUEST FOR I/O?
8326 035736 002403 BLT 25 ;YES--BRANCH
8327 035740 004737 036272 JSR PC, C14 ;CALL THE COMMAND INITIATOR
8328 035744 000440 BR 85 ;BRANCH TO EXIT
8329 035746 005737 034502 25: TST DTUW ;DATA TRANSFER UNDERWAY?
8330 035752 002012 BGE 45 ;YES--GO START A SEARCH
8331 035754 005737 034460 TST SEEKFG ;DO IMPLIED SEEKS?
8332 035760 100404 BMI 35 ;YES---BRANCH
8333 035762 004037 037242 JSR RO, LA ;NO--DO LOOK AHEAD
8334 035766 000427 BR 85 ;RETURN HERE ON A PARITY ERROR
8335 035770 000403 BR 45 ;GO START A SEARCH
8336 035772 004737 036056 35: JSR PC, C11 ;START A DATA TRANSFER
8337 035776 000423 BR 85
8338 036000 004737 036164 45: JSR PC, C13 ;START A SEARCH
8339 036004 000420 BR 85 ;GO TO THE EXIT
8340 036006 112761 177777 034420 55: MOVB #-1, DPRQS(R1) ;SET PORT REQUEST INDICATOR
8341 036014 010103 MOV R1, R3 ;SET UP TO ADDRESS WORDS
8342 036016 006303 ASL R3 ;CONVERT TO WORD INDEX
8343 036020 012763 023420 034462 MOV #10000, TIMER(R3) ;START 10 SEC TIMER
8344 036026 000402 BR 75 ;EXIT
8345 036030 004737 036706 65: JSR PC, C17 ;PROCESS THE PARITY ERROR
8346 036034 032714 000100 75: BIT #BIT06, (R4) ;SEE IF 'IE' ALREADY SET
8347 036040 001002 BNE 85 ;BR IF SET
8348 036042 004737 042154 JSR PC, SET. IE ;SET "IE" WITHOUT A "TRE"
8349 036046 012637 177776 85: MOV (SP)+, @#PS ;RESTORE PROC. STATUS
8350 036052 104413 RESREG ;RESTORE R0 - R5
8351 036054 000207 RTS PC
8352 ;
8353 ;COMMAND INITIATOR
    
```



```

8354
8355 ;CALL
8356 ;MOV #DRVNUM,R1 ;DRIVE NUMBER
8357 ;MOV #DPB,R2 ;ADDRESS OF DPB
8358 ;JSR PC,C1? ;C1?= C11,C13, OR C14
8359 ;WHERE:
8360 ;C11=DATA TRANSFER
8361 ;C12=SEARCH REQUESTED BY DATA XFER
8362 ;C14=NOT DATA TRANSFER
8363
8364 036056 004737 042714 C11: JSR PC,POPQUE ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
8365 036062 010237 034430 MOV R2,TRNSWT ;PUT REQ. IN TRANSFER WAIT QUEUE
8366 036066 010203 MOV R2,R3 ;DPB ADDRESS TO R3
8367 036070 013704 034516 MOV RPADR,R4 ;RPCS1 ADDRESS
8368 036074 010164 000010 MOV R1,RPCS2(R4) ;SELECT DRIVE
8369 036100 062703 000004 ADD #4,R3 ;DESIRED WORD COUNT
8370 036104 062704 000002 ADD #2,R4 ;RPWC ADDRESS
8371 036110 012324 MOV (R3)+,(R4)+ ;LOAD WORD COUNT
8372 036112 012324 MOV (R3)+,(R4)+ ;LOAD BUFFER ADDRESS
8373 036114 012346 MOV (R3)+,-(SP) ;LOAD SECTOR AND TRACK
8374 036116 004037 041644 JSR RO,WRT.RP ;CALL THE LOAD(WRITE) ROUTINE
8375 036122 000006 RPDA ;INDEX OF REGISTER TO LOAD
8376 036124 036706 C17 ;ERROR RETURN ADDRESS
8377 036126 012346 MOV (R3)+,-(SP) ;LOAD CYLINDER ADDRESS
8378 036130 004037 041644 JSR RO,WRT.RP
8379 036134 000034 RPCA
8380 036136 036706 C17
8381 036140 016246 000002 MOV 2(R2),-(SP) ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
8382 036144 004037 041644 JSR RO,WRT.RP
8383 036150 000000 RPCS1
8384 036152 036706 C17
8385 036154 010137 034502 MOV R1,DTUW ;SET "DATA TRANSFER UNDERWAY"
8386 036160 000137 036650 JMP C15
8387 036164 013704 034516 C13: MOV RPADR,R4 ;RPCS1 ADDRESS
8388 036170 010164 000010 MOV R1,RPCS2(R4) ;SELECT DRIVE
8389 036174 016246 000012 MOV 12(R2),-(SP) ;DESIRED CYLINDER ADDRESS
8390 036200 004037 041644 JSR RO,WRT.RP
8391 036204 000034 RPCA
8392 036206 036706 C17
8393 036210 116203 000010 MOV#B 10(R2),R3 ;PICKUP SECTOR ADDRESS
8394 036214 163703 034532 SUB MXWINDW,R3 ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
8395 036220 002002 BGE 1$
8396 036222 062703 000026 ADD #22,R3
8397 036226 010346 15: MOV R3,-(SP) ;COMBINE THE ADJUSTED SECTOR WITH
8398 036230 116266 000011 000001 MOV#B 11(R2),1(SP) ;THE DESIRED TRACK
8399 036236 004037 041644 JSR RO,WRT.RP ;LOAD DESIRED TRACK & SECTOR
8400 036242 000006 RPDA
8401 036244 036706 C17
8402 036246 012746 000131 MOV #131,-(SP) ;START A SEARCH
8403 036252 004037 041644 JSR RO,WRT.RP
8404 036256 000000 RPCS1
8405 036260 036706 C17
8406 036262 156137 034504 034432 B1SB ATABIT(R1),SRCHWT ;SET "SEARCH WAIT" KEY
8407 036270 000567 BR C15
8408 036272 013704 034516 C14: MOV RPADR,R4 ;RPCS1 ADDRESS
8409 036276 010164 000010 MOV R1,RPCS2(R4) ;SELECT DRIVE
  
```



8410	036302	116203	000002		MOVB	2(R2),R3	; PICKUP THE REQUESTED COMMAND
8411	036306	122703	000131		CMPB	#131,R3	; IS IT A SEARCH COMMAND?
8412	036312	001007			BNE	15	; BRANCH IF NO
8413	036314	016246	000010		MOV	10(R2),-(SP)	; LOAD DESIRED TRACK & SECTOR
8414	036320	004037	041644		JSR	RO,WRT.RP	
8415	036324	000006			RPDA		
8416	036326	036706			C17		
8417	036330	000403			BR	25	; GO LOAD CYLINDER
8418	036332	122703	000105	15:	CMPB	#105,R3	; IS IT A SEEK COMMAND
8419	036336	001007			BNE	35	; BRANCH IF NO
8420	036340	016246	000012	25:	MOV	12(R2),-(SP)	; LOAD DESIRED CYLINDER
8421	036344	004037	041644		JSR	RO,WRT.RP	
8422	036350	000034			RPCA		
8423	036352	036706			C17		
8424	036354	000546			BR	C16	
8425	036356	122703	000115	35:	CMPB	#115,R3	; IS IT AN "OFFSET" COMMAND?
8426	036362	001013			BNE	45	; BR IF NO
8427	036364	004037	041464		JSR	RO,RD.RP	; MERGE THE OFFSET VALUE INTO RPOF
8428	036370	000032			RPOF		; BUT DON'T CHANGE THE UPPER
8429	036372	036706			C17		
8430	036374	116216	000001		MOVB	1(R2),(SP)	; BYTE WHEN LOADING THE
8431	036400	004037	041644		JSR	RO,WRT.RP	; REGISTER (RPOF)
8432	036404	000032			RPOF		
8433	036406	036706			C17		
8434	036410	000530			BR	C16	; GO START THE COMMAND
8435	036412	122703	000107	45:	CMPB	#107,R3	; IS IT A "RECALIBRATE" COMMAND?
8436	036416	001525			BEQ	C16	; BRANCH IF YES
8437	036420	122703	000117		CMPB	#117,R3	; IS IT A RETURN TO CENTER?
8438	036424	001522			BEQ	C16	; BRANCH IF YES
8439	036426	122703	000103		CMPB	#103,R3	; IS IT AN "UNLOAD" COMMAND?
8440	036432	001016			BNE	55	; BRANCH IF NO
8441	036434	112761	000001	034360	MOVB	#1,DRVACT(R1)	; SET THE DRIVE ACTIVE INDICATOR
8442	036442	105061	034370		CLRB	DRVSTA(R1)	; PUT DRIVE STATUS TO OFFLINE
8443	036446	112761	000001	034436	MOVB	#1,ULDFLG(R1)	; SET "UNLOAD IN PROGRESS" FLAG
8444	036454	010346			MOV	R3,-(SP)	; START THE "UNLOAD" COMMAND
8445	036456	004037	041644		JSR	RO,WRT.RP	
8446	036462	000000			RPCS1		
8447	036464	036706			C17		
8448	036466	000207			RTS	PC	; RETURN TO USER
8449	036470	122703	000143	55:	CMPB	#143,R3	; IS IT A "SET FORMAT" COMMAND?
8450	036474	001014			BNE	65	; BRANCH IF NO
8451	036476	004037	041464		JSR	RO,RD.RP	; READ THE OFFSET REGISTER
8452	036502	000032			RPOF		
8453	036504	036706			C17		
8454	036506	116266	000001	000001	MOVB	1(R2),1(SP)	; COMBINE "FMT22", "ECI", AND "HCI"
8455	036514	004037	041644		JSR	RO,WRT.RP	; LOAD "FMT22", "ECI", AND/OR "HCI".
8456	036520	000032			RPOF		
8457	036522	036706			C17		
8458	036524	000436			BR	125	
8459	036526	122703	000141	65:	CMPB	#141,R3	; IS IT A "GET REGISTER" COMMAND?
8460	036532	001023			BNE	105	; BRANCH IF NO
8461	036534	016203	000006	75:	MOV	6(R2),R3	; POINTS TO 1ST ADDRESS OF WHERE
8462							; TO PUT THE REGISTER(S)
8463	036540	116237	000010	036556	MOVB	10(R2),95	; INIT. THE INDEX FOR THE FIRST REG.
8464	036546	116205	000011		MOVB	11(R2),R5	; INDEX OF LAST REG. TO MOVE
8465	036552	004037	041464	85:	JSR	RO,RD.RP	; READ RPO4/5/6 REGISTER



```

8466 036556 000000          95:  RPCS1          ; INDEX OF REG. TO READ
8467 036560 036706          C17
8468 036562 012623          MOV (SP)+, (R3)+ ; GET THE CONTENTS OF RH11/RPO4/5/6 REG.
8469 036564 023705 036556  CMP 95, R5        ; LAST REG. BEEN READ?
8470 036570 001414          BEQ 125          ; GET OUT IF YES
8471 036572 062737 000002 036556  ADD #2, 95       ; INCREASE THE INDEX BY 2
8472 036600 000764          BR 85           ; LOOP--MORE TO READ
8473 036602 122703 000145          105: CMPB #145, R3    ; IS IT A "SELECT DRIVE" COMMAND?
8474 036606 001405          BEQ 125          ; BRANCH IF YES
8475 036610 010346          115: MOV R3, -(SP)    ; LOAD THE COMMAND
8476 036612 004037 041644          JSR RO, WRT. RP
8477 036616 000000          RPCS1
8478 036620 036706          C17
8479 036622 004737 042714          125: JSR PC, POPQUE   ; REMOVE REQ. FROM QUEUE
8480 036626 052762 000200 000016  BIS #BIT07, 16(R2) ; SET THE "DONE" BIT
8481 036634 005737 034456          TST SAVEFG      ; SAVE THE RH11/RPO4/5/6 REGISTERS?
8482 036640 100002          BPL 135         ; BRANCH IF NO
8483 036642 004737 042036          JSR PC, SVRH11  ; YES--GO SAVE THE REGISTERS
8484 036646 000207          135: RTS PC         ; RETURN TO USER
8485 036650 006301          C15: ASL R1
8486 036652 012761 001750 034462  MOV #1000., TIMER(R1) ; SET A ONE SECOND TIMER
8487 036660 006201          ASR R1
8488 036662 112761 000001 034360  MOVB #1, DRVACT(R1) ; SET THE DRIVE ACTIVE
8489 036670 000207          RTS PC         ; RETURN TO THE USER
8490 036672 010346          C16: MOV R3, -(SP)    ; LOAD THE COMMAND
8491 036674 004037 041644          JSR RO, WRT. RP
8492 036700 000000          RPCS1
8493 036702 036706          C17
8494 036704 000761          BR C15
8495 036706 032764 010000 000010  C17: BIT #BIT12, RPCS2(R4) ; DRIVE NON-EXISTENT ?
8496 036714 001034          BNE C18        ; BR IF YES
8497 036716 005702          15: TST R2        ; ANYTHING IN QUEUE ?
8498 036720 001405          BEQ C17B       ; BR IF NOT
8499 036722 012762 104000 000016  MOV #BIT15!BIT11, 16(R2) ; SET "PARITY" ERROR INDICATOR
8500 036730 004737 042036          JSR PC, SVRH11 ; GO SAVE THE RH11/RPO4/5/6 REGISTERS
8501 036734 012746 000111          C17B: MOV #111, -(SP) ; DO A "DRIVE CLEAR"
8502 036740 004037 041644          JSR RO, WRT. RP
8503 036744 000000          RPCS1
8504 036746 037006          C18
8505 036750 004737 042576          JSR PC, EMPTYQ  ; EMPTY THE QUEUE
8506 036754 105061 034436          CLRB ULDFLG(R1) ; CLEAR THE UNLOAD IN QUEUE FLAG
8507 036760 105061 034360          CLRB DRVACT(R1) ; DRIVE IS IDLE
8508 036764 020137 034502          CMP R1, DTUW   ; IF THIS DRIVE HAD AN I/O REQUEST
8509 036770 001005          BNE 15        ; IN PROGRESS CLEAR ALL OF THE FLAGS
8510 036772 005037 034430          CLR TRNSWT
8511 036776 012737 177777 034502  MOV #-1, DTUW
8512 037004 000207          15: RTS PC
8513 037006 104412          C18: SAVREG      ; SAVE R0 - R5
8514 037010 032764 010000 000010  BIT #BIT12, RPCS2(R4) ; IS 'NED' SET ?
8515 037016 001002          BNE 15        ; BR IF YES
8516 037020 005001          CLR R1
8517 037022 005003          CLR R3
8518 037024 105761 034360          15: TSTB DRVACT(R1) ; DRIVE ACTIVE?
8519 037030 001443          BEQ 55        ; BRANCH IF NO
8520 037032 013702 034430          MOV TRNSWT, R2 ; GET THE "TRANSFER WAIT" QUEUE
8521 037036 020137 034502          CMP R1, DTUW   ; DID THIS DRIVE HAVE AN I/O IN PROGRESS?
  
```



```

8522 037042 001402          BEQ      25          ;BRANCH IF YES
8523 037044 004737 042672   JSR      PC,GETREQ  ;GET THE DPB POINTER
8524 037050 005702          TST      R2          ;QUEUE ENTRY FOR DRIVE ?
8525 037052 001415          BEQ      45          ;BR IF NOT
8526 037054 032764 010000 000010 BIT      #BIT12,RPCS2(R4) ;'MED' SET ?
8527 037062 001404          BEQ      35          ;BR IF NOT
8528 037064 012762 100002 000016 MOV      #BIT15!BIT01,16(R2) ;SET 'DRIVE NON-EXISTENT' INDICATOR
8529 037072 000405          BR       45          ;CONTINUE
8530 037074 012762 102000 000016 35: MOV      #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
8531 037102 004737 042036   JSR      PC,SVRH11  ;SAVE RH11/RPO4/5/6 REGISTERS
8532 037106 012763 177777 034462 45: MOV      #-1,TIMER(R3) ;STOP THE TIMER
8533 037114 105061 034360   CLRB    DRVACT(R1)  ;SET "DRIVE ACTIVE" TO IDLE
8534 037120 020137 034502   CMP      R1,DTUW    ;IS THIS DRIVE SETUP FOR A TRANSFER
8535 037124 001005          BNE     55          ;BR IF NOT
8536 037126 012737 177777 034502 MOV      #-1,DTUW   ;RESET THE INDICATOR
8537 037134 005037 034430   CLR     TRNSWT     ;CLEAR THE TRANSFER QUEUE
8538 037140 105061 034436   CLRB    ULDFLG(R1) ;CLEAR UNLOAD FLAG
8539 037144 032764 010000 000010 55: BIT      #BIT12,RPCS2(R4) ;'MED' SET ?
8540 037152 001021          BNE     65          ;BR IF YES
8541 037154 005201          INC     R1          ;MOVE TO THE NEXT DRIVE
8542 037156 062703 000002   ADD     #2,R3
8543 037162 042701 177770   BIC     #C7,R1
8544 037166 001316          BNE     15          ;BRANCH IF MORE DRIVES
8545 037170 012737 177777 034502 MOV      #-1,DTUW   ;NO DATA TRANSFERS UNDERWAY
8546 037176 005037 034430   CLR     TRNSWT     ;CLEAR THE 'TRANSFER WAIT' QUEUE
8547 037202 004737 042520   JSR     PC,CLRQUE  ;CLEAR ALL OF THE REQUEST QUEUES
8548 037206 012764 000040 000010 MOV      #BIT05,RPCS2(R4) ;DO A MASSBUS INIT.
8549 037214 000406          BR       75          ;CONTINUE
8550 037216 004737 042576   JSR     PC,EMPTYQ  ;CLEAR THE DRIVE'S QUEUE
8551 037222 105061 034370   CLRB    DRVSTA(R1) ;SET DRIVE TO OFFLINE
8552 037226 105061 034400   CLRB    DRVTYP(R1) ;CLEAR THE DRIVE TYPE INDICATOR
8553 037232 004737 042154   JSR     PC,SET. IE ;SET "IE" WITHOUT "TRE"
8554 037236 104413          RESREG  ;RESTORE R0 - R5
8555 037240 000207          RTS     PC          ;RETURN
8556
8557          ;LOOK AHEAD ROUTINE
8558          ;
8559          ;CALL
8560          ;
8561          ;   MOV      #DRVNUM,R1      ;DRIVE NUMBER
8562          ;   MOV      #DPB,R2       ;POINT TO DPB
8563          ;   JSR      RO,LA       ;GO CHECK THE WINDOW
8564          ;   RETURN1     ;ERROR RETURN
8565          ;   RETURN2     ;START A SEARCH
8566          ;   RETURN3     ;START A DATA TRANSFER
8567 037242 013704 034516   LA:    MOV      RPADR,R4      ;GET RPCS1'S ADDRESS
8568 037246 010164 000010   MOV      R1,RPCS2(R4)      ;SELECT DRIVE
8569 037252 004037 041464   JSR      RO,RO.RP         ;READ CURRENT CYLINDER
8570 037256 000036          RPCC
8571 037260 037372          45          ;ERROR RETURN ADDRESS
8572 037262 022662 000012   CMP      (SP)+,12(R2)      ;IS CURRENT CYLINDER=DESIRED
8573          ;CYLINDER?
8574 037266 001037          BNE     35          ;EXIT IF NO
8575 037270 105261 034446   INCB    LACNT(R1)         ;INCREMENT THE LOOK AHEAD COUNT
8576 037274 126137 034446 034524 CMPB    LACNT(R1),MXLACT  ;EXCEED MAX?
8577 037302 003026          BGT     25          ;BRANCH IF YES
    
```



```

8578 037304 116203 000010          MOVB    10(R2),R3      ;GET DESIRED SECTOR ADDRESS AND
8579 037310 000303                SWAB    R3            ;MULT. BY 64--ALIGN WITH
8580 037312 006203                ASR     R3            ;LOOK AHEAD REGISTER
8581 037314 006203                ASR     R3
8582 037316 012737 000340 177776    MOV     #340,2#PS     ;PRIORITY LEVEL "7"
8583 037324 004037 041464          JSR     RO,RD.RP      ;READ LOOK AHEAD REGISTER
8584 037330 000020                RPLA
8585 037332 037372                4$
8586 037334 162603                SUB     (SP)+,R3     ;CALCULATE THE DELTA
8587 037336 002002                BGE    1$
8588 037340 062703 002600          ADD     #(<22.*64.>),R3 ;MAKE THE DELTA POSITIVE
8589 037344 023703 034526          1$:    CMP     MXDLTA,R3  ;CHECK THE DELTA TO SEE
8590 037350 002406                BLT    3$            ;IF IT IS WITHIN THE
8591 037352 023703 034530          CMP     MNDLTA,R3   ;WINDOW---IF YES, ZERO
8592 037356 002003                BGE    3$            ;THE LOOK AHEAD COUNT
8593 037360 105061 034446          2$:    CLRB   LACNT(R1) ;AND TAKE THE I/O EXIT
8594 037364 005720                TST    (RO)+
8595 037366 005720          3$:    TST    (RO)+   ;ADJUST THE RETURN ADDRESS
8596 037370 000402                BR     5$            ;EXIT
8597 037372 004737 036706          4$:    JSR     PC,C17  ;PROCESS THE ERROR
8598 037376 000200          5$:    RTS     RO      ;RETURN
8599
8600                                ; INTERRUPT SERVICE ROUTINE
8601
8602 037400 112737 000001 034434  ISR:  MOVB    #1,ACTDRV     ;SET "ACTIVE DRIVER" FLAG
8603 037406 104412                SAVREG
8604 037410 013704 034516          MOV     RPADR,R4     ;ADDRESS OF RHSCS1
8605 037414 013701 034502          MOV     DTUW,R1      ;GET "DATA TRANSFER UNDERWAY" INDICATOR
8606 037420 002403                BLT    1$            ;BRANCH IF NO DATA TRANSFER UNDERWAY
8607 037422 004737 037444          JSR     PC,TD        ;CALL TRANSFER DONE
8608 037426 000402                BR     2$            ;EXIT
8609 037430 004737 037604          1$:    JSR     PC,SC   ;CALL SPECIAL CONDITIONS
8610 037434 104413                2$:    RESREG
8611 037436 105037 034434          CLRB   ACTDRV       ;CLEAR "ACTIVE DRIVER" FLAG
8612 037442 000002                RTI
8613
8614                                ; TRANSFER DONE ROUTINE
8615
8616 037444 105061 034360          TD:    CLRB   DRVACT(R1) ;SET DRIVE ACTIVE INDICATOR TO IDLE
8617 037450 012737 177777 034502    MOV     #-1,DTUW     ;NO DATA TRANSFERS UNDERWAY
8618 037456 006301                ASL    R1
8619 037460 012761 177777 034462    MOV     #-1,TIMER(R1) ;CANCEL TIMEOUT
8620 037466 006201                ASR    R1
8621 037470 013702 034430          MOV     TRNSWT,R2   ;GET "DPB" ADDRESS FROM THE
8622 037474 005037 034430          CLR    TRNSWT       ;TRANSFER WAIT QUEUE--CLEAR QUEUE
8623 037500 052762 000200 000016    BIS    #BIT07,16(R2) ;SET DONE
8624 037506 010164 000010          MOV     R1,RPCS2(R4) ;SELECT THE DRIVE
8625 037512 004037 041464          JSR     RO,RD.RP     ;TRANSFER ERROR(TRE=1)?
8626 037516 000000                RPCS1
8627 037520 036706                C17
8628 037522 006126                ROL    (SP)+
8629 037524 100413                BMI    3$            ;BR IF YES
8630 037526 005737 034456          TST    SAVEFG       ;SAVE THE RH11/RPO4/5/6 REGISTERS?
8631 037532 100002                BPL    1$            ;BRANCH IF NO
8632 037534 004737 042036          JSR     PC,SVRH11   ;YES--SAVE THE REGISTERS
8633 037540 004737 035576          1$:    JSR     PC,OPT  ;CALL OPTIMIZER
    
```

```

8634 037544 000417          BR      SC      ;CHECK OTHER DRIVES
8635 037546 012714 000113 25:     MOV     #113, (R4) ;RELEASE THE DRIVE
8636 037552 000414          BR      SC      ;CHECK FOR OTHER DRIVES
8637 037554 052762 100100 000016 35:     BIS     #BIT15!BIT06, 16(R2) ;SET DATA ERROR FLAG
8638 037562 004737 042576          JSR     PC, EMPTYQ ;EMPTY THE "DRIVE'S WAIT" QUEUE
8639 037566 004737 042036          JSR     PC, SVRH11 ;SAVE THE RH11/RPO4/5/6 REGISTERS
8640 037572 012714 040111          MOV     #40111, (R4) ;ISSUE A "DRIVE CLEAR"
8641 037576 012714 000113          MOV     #113, (R4) ;ISSUE A RELEASE TO THE DRIVE
8642 037602 000400          BR      SC      ;CHECK FOR OTHER DRIVES
8643
8644          ;SPECIAL CONDITION ROUTINE
8645
8646 037604 116403 000016  SC:     MOVVB  RPAS(R4), R3 ;READ "RPAS"
8647 037610 001014          BNE     25 ;BRANCH IF ANY 'ATA' BITS SET
8648 037612 004037 041464          JSR     RO, RD. RP ;READ CONTROL AND STATUS REGISTER
8649 037616 000000          RPCS1
8650 037620 037006          CIB
8651 037622 106126          ROLB   (SP)+ ;IS "IE"=1?
8652 037624 100405          BMI     15 ;YES, NO DRIVES TO CHECK
8653 037626 004037 042756          JSR     RO, ES. SAV ;SAVE THE ADDRESS IN 'SESCAPE'
8654 037632 104001          ERROR  1 ;REPORT AN ILLEGAL INTERRUPT
8655 037634 004737 042154          JSR     PC, SET. IE ;SET INTERRUPT ENABLE
8656 037640 000207          15:     RTS     PC ;RETURN
8657 037642 005046          25:     CLR     -(SP) ;PROCESS ALL DRIVES THAT HAVE
8658 037644 110316          MOVVB  R3, (SP) ;AN "ATA"=1
8659 037646 012703 000001          MOV     #1, R3
8660 037652 005001          CLR     R1
8661 037654 030316  SC3:    BIT     R3, (SP) ;ATA=1?
8662 037656 001005          BNE     SC5 ;YES--BRANCH
8663 037660 005201  SC4:    INC     R1 ;MOVE TO THE NEXT DRIVE
8664 037662 106303          ASLB   R3
8665 037664 001373          BNE     SC3 ;BRANCH IF MORE TO CHECK?
8666 037666 005726          TST    (SP)+ ;CLEAN OFF THE STACK
8667 037670 000207          RTS     PC ;RETURN TO USER
8668 037672 105761 034410  SC5:    TSTB   DPINT(R1) ;INITIALIZING THE DRIVE ?
8669 037676 001402          BEQ     15 ;BR IF NOT
8670 037700 000137 040576          JMP     SC13 ;PROCESS THE DRIVE
8671 037704 105761 034420  15:     TSTB   DPRQS(R1) ;PORT REQUEST OUTSTANDING ?
8672 037710 001402          BEQ     25 ;BR IF NOT
8673 037712 000137 040576          JMP     SC13 ;START THE OUTSTANDING COMMAND
8674 037716 105761 034370  25:     TSTB   DRVSTA(R1) ;CHECK THE DRIVE STATUS
8675 037722 003025          BGT     55 ;BRANCH IF ONLINE
8676 037724 105761 034436          TSTB   ULDFLG(R1) ;UNLOAD IN PROGRESS?
8677 037730 003422          BLE     55 ;BRANCH IF NOT
8678 037732 004737 042672          JSR     PC, GETREQ ;GET DPB POINTER
8679 037736 004737 042036          JSR     PC, SVRH11 ;SAVE THE RH11/RPO4/5/6 REGISTERS
8680 037742 004737 040526          JSR     PC, SC12 ;SAVE RPOS1, RPER1, RPER2, AND RPER3
8681
8682 037746 105761 034370          TSTB   DRVSTA(R1) ;DID DRIVE COME ONLINE?
8683 037752 003416          BLE     65 ;NO---BRANCH
8684 037754 032737 040000 034350  BIT     #BIT14, RPERRS ;WAS THERE AN ERROR?
8685 037762 001002          BNE     35 ;BR IF ERROR
8686 037764 000137 040436          JMP     SC11 ;NO ERROR
8687 037770 013705 034352  35:     MOV     RPERRS+2, R5 ;YES -- PICKUP RPER1 AND
8688 037774 000502          BR      SC6A ;GO PROCESS THE ERROR
8689 037776 105761 034360  55:     TSTB   DRVACT(R1) ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
    
```



```

8690 040002 001033          BNE      SC6          ;BR IF EITHER
8691 040004 004737 040526   JSR      PC,SC12     ;SAVE RPDS1, RPER1, RPER2, AND RPER3
8692                                ;ALSO DO A DRVINT
8693 040010 105761 034410   65:     TSTB     DPINT(R1) ;TRYING TO INIT THE DRIVE ?
8694 040014 001321          BNE      SC4          ;BR IF YES, CHECK ON MORE DRIVES
8695 040016 105761 034370   TSTB     DRVSTA(R1) ;CHECK ON DRIVE'S STATUS
8696 040022 100412          BMI      75          ;BR IF UNSAFE
8697 040024 032737 020000 034356 BIT      #BIT13,RPERRS+6 ;ADDRESS PLUG CHANGED ?
8698 040032 001013          BNE      85          ;BR IF YES
8699 040034 012746 000113   MOV      #113,-(SP)  ;RELEASE COMMAND
8700 040040 004037 041644   JSR      RO,WRT.RP  ;WRITE THE COMMAND INTO RPCS1
8701 040044 000000          RPCS1          ;REGISTER INDEX
8702 040046 040406          SC8          ;PARITY EXIT ADDRESS
8703 040050 011605          75:     MOV      (SP),R5 ;PICKUP (RPAS) BEFORE THE ERROR CALL
8704 040052 004037 042756   JSR      RO,ES.SAV  ;SAVE THE ADDRESS IN 'SESCAPE'
8705 040056 104002          ERROR      2        ;REPORT THE UNEXPECTED ATTENTION
8706 040060 000677          BR        SC4        ;GO CHECK FOR MORE ATA'S
8707 040062
8708 040062 004037 042756   85:     JSR      RO,ES.SAV ;SAVE THE ADDRESS IN 'SESCAPE'
8709 040066 104005          ERROR      5        ;REPORT THE ADDRESS PLUG CHANGE
8710 040070 000673          BR        SC4        ;CHECK FOR MORE DRIVES
8711 040072 006301          SC6:     ASL      R1        ;SETUP TO ADDRESS WORDS
8712 040074 012761 177777 034462 MOV      #-1,TIMER(R1) ;STOP THE TIMER
8713 040102 006201          ASR      R1        ;RESTORE THE DRIVE ADDRESS
8714 040104 004737 042672   JSR      PC,GETREQ  ;GET THE DPB POINTER FROM THE QUEUE
8715 040110 010164 000010   MOV      R1,RPCS2(R4) ;SELECT DRIVE
8716 040114 004037 041464   JSR      RO,RD.RP  ;READ THE RPO4'S STATUS REG.
8717 040120 000012          RPDS1
8718 040122 040406          SC8
8719 040124 011605          MOV      (SP),R5    ;AND PUT IT IN R5
8720 040126 006126          ROL      (SP)+      ;WAS THERE AN ERROR?
8721 040130 100407          BMI      15        ;BR IF ERROR
8722 040132 105761 034360   TSTB     DRVACT(R1) ;CHECK DRIVE'S STATE
8723 040136 003137          BGT      SC11      ;BR IF DRIVE ACTIVE WITH ORDER
8724 040140 052762 100210 000016 BIS      #BIT15!BIT07!BIT03,16(R2) ;INFORM USER OF ERROR RECOVER COMPLETION
8725 040146 000470          BR        SC7
8726 040150 004037 041464   15:     JSR      RO,RD.RP ;READ ERROR REGISTER #1
8727 040154 000014          RPER1
8728 040156 040406          SC8
8729 040160 012605          MOV      (SP)+,R5  ;AND SAVE IT IN R5
8730 040162 004737 042036   JSR      PC,SVRH11 ;SAVE RH11/RPO4/5/6 REGISTERS
8731 040166 012746 000111   MOV      #111,-(SP) ;ISSUE A DRIVE CLEAR
8732 040172 004037 041644   JSR      RO,WRT.RP
8733 040176 000000          RPCS1
8734 040200 040406          SC8
8735 040202 006105          SC6A:    ROL      R5        ;WAS "UNSAFE" CONDITION =1?
8736 040204 100406          BMI      15        ;BRANCH IF YES
8737 040206 005702          TST     R2        ;ANYTHING IN QUEUE ?
8738 040210 001447          BEQ     SC7        ;BR IF NOT
8739 040212 052762 100240 000016 BIS      #BIT15!BIT07!BIT05,16(R2) ;INFORM USER OF ERROR
8740 040220 000443          BR        SC7
8741 040222 004037 041464   15:     JSR      RO,RD.RP ;READ DRIVE STATUS REG. #1
8742 040226 000012          RPDS1
8743 040230 040406          SC8
8744 040232 011605          MOV      (SP),R5  ;SAVE RPDS1 IN R5
8745 040234 006126          ROL      (SP)+      ;"ERR"=1?
    
```

8746	040236	100011			BPL	25		;BR IF NO--UNSAFE CLEARED
8747	040240	112761	177777	034370	MOVB	#-1,DRVSTA(R1)		;DRIVE IS UNSAFE
8748	040246	004737	042036		JSR	PC,SVRH11		;SAVE RH11/RPO4/5/6 REGISTERS
8749	040252	052762	110000	000016	BIS	#BIT15!BIT12,16(R2)		;INFORM USER OF UNSAFE ERROR
8750	040260	000423			BR	SC7		
8751	040262	032705	010000		25:	BIT	#BIT12,R5	;"MOL" = 1 ?
8752	040266	001015			BNE	35		;BR IF YES
8753	040270	112761	177777	034360	MOVB	#-1,DRVACT(R1)		;ACTIVE ERROR RECOVER
8754	040276	112761	000001	034370	MOVB	#1,DRVSTA(R1)		;ONLINE
8755	040304	006301			ASL	R1		
8756	040306	012761	072460	034462	MOV	#30000.,TIMER(R1)		;START 30 SECOND TIMER
8757	040314	006201			ASR	R1		
8758	040316	000137	037660		JMP	SC4		
8759	040322	052762	100220	000016	35:	BIS	#BIT15!BIT07!BIT04,16(R2)	;INFORM USER OF ERROR
8760	040330	105061	034360		SC7:	CLRB	DRVACT(R1)	;DRIVE IS IDLE
8761	040334	004737	042576		JSR	PC,EMPTYQ		;DUMP THE QUEUE
8762	040340	105761	034436		TSTB	ULDFLG(R1)		;UNLOAD IN PROGRESS OR QUEUE?
8763	040344	003002			BGT	15		;BR IF NOT
8764	040346	105061	034436		CLRB	ULDFLG(R1)		;CLEAR UNLOAD FLAG
8765	040352	116164	034504	000016	15:	MOVB	ATABIT(R1),RPAS(R4)	;CLEAR ATTENTION BIT
8766	040360	105761	034370		TSTB	DRVSTA(R1)		;IS THE DRIVE UNSAFE ?
8767	040364	100406			BMI	25		;BR IF IT IS
8768	040366	012746	000113		MOV	#113,-(SP)		;RELEASE COMMAND
8769	040372	004037	041644		JSR	RO,WRT.RP		;WRITE THE COMMAND INTO RPCS1
8770	040376	000000			RPCS1			;REGISTER INDEX
8771	040400	040406			SC8			;PARITY EXIT ADDRESS
8772	040402	000137	037660		25:	JMP	SC4	;CHECK FOR MORE DRIVES
8773	040406	105761	034360		SC8:	TSTB	DRVACT(R1)	;IS DRIVE IDLE?
8774	040412	001405			BEQ	15		;YES--BRANCH
8775	040414	004737	042672		JSR	PC,GETREQ		;GET DPB POINTER
8776	040420	004737	036706		JSR	PC,C17		;PROCESS THE PARITY ERROR
8777	040424	000402			BR	25		;CONTINUE
8778	040426	004737	036734		15:	JSR	PC,C17B	;PROCESS THE UNCORRECTABLE PARITY ERROR
8779	040432	000137	037660		25:	JMP	SC4	;CHECK MORE DRIVES
8780	040436	105761	034436		SC11:	TSTB	ULDFLG(R1)	;"UNLOAD IN PROGRESS"?
8781	040442	003402			BLE	15		;BRANCH IF NO
8782	040444	105061	034436		CLRB	ULDFLG(R1)		;CLEAR UNLOAD FLAG
8783	040450	105061	034360		15:	CLRB	DRVACT(R1)	;SET DRIVE IDLE
8784	040454	136137	034504	034432	BITB	ATABIT(R1),SRCHWT		;DOING A SEARCH OPERATION FOR
8785								;AN I/O COMMAND?
8786	040462	001012			BNE	25		;BRANCH IF YES
8787	040464	004737	042714		JSR	PC,POPQUE		;REMOVE REQUEST FROM QUEUE
8788	040470	052762	000200	000016	BIS	#BIT07,16(R2)		;SET "DONE" BIT
8789	040476	005737	034456		TST	SAVEFG		;SAVE THE REGISTERS?
8790	040502	100002			BPL	25		;BRANCH IF NO
8791	040504	004737	042036		JSR	PC,SVRH11		;YES--SAVE ALL OF THE RH11/RPO4/5/6 REG'S
8792	040510	116164	034504	000016	25:	MOVB	ATABIT(R1),RPAS(R4)	;CLEAR ATTENTION BIT
8793	040516	004737	035576		JSR	PC,OPT		;START A REQUEST
8794	040522	000137	037660		JMP	SC4		;CHECK FOR MORE DRIVES
8795	040526	010164	000010		SC12:	MOV	R1,RPCS2(R4)	;SELECT DRIVE
8796	040532	016437	000012	034350	MOV	RPDS1(R4),RPERRS		;SAVE THE FOUR REGISTERS THAT
8797	040540	016437	000014	034352	MOV	RPER1(R4),RPERRS+2		;WILL TELL US SOMETHING
8798	040546	016437	000040	034354	MOV	RPER2(R4),RPERRS+4		
8799	040554	016437	000042	034356	MOV	RPER3(R4),RPERRS+6		
8800	040562	004037	034746		JSR	RO,DRVINT		;INIT. THE STATE OF THE DRIVE
8801	040566	000401			BR	15		;TAKE ERROR EXIT



```

8802 040570 000207          RTS      PC          ;RETURN
8803 040572 005726          1$:     TST      (SP)+    ;POP PC OFF OF THE STACK
8804 040574 000704          BR       SC8         ;PROCESS THE PARITY ERROR
8805 040576 006301          SC13:  ASL      R1         ;SETUP TO ADDRESS WORDS
8806 040600 012761 177777 034462 MOV     #-1,TIMER(R1) ;STOP THE TIMER
8807 040606 006201          ASR      R1         ;
8808 040610 010164 000010     MOV     R1,RPCS2(R4) ;SELECT THE DRIVE
8809 040614 116164 034504 000016 MOVB    ATABIT(R1),RPAS(R4) ;CLEAR THE ATTENTION BIT
8810 040622 032714 004000     BIT     #BIT11,(R4)  ;DRIVE AVAILABLE ?
8811 040626 001006          BNE     1$         ;BR IF AVAILABLE
8812 040630 006301          ASL      R1         ;
8813 040632 012761 023420 034462 MOV     #10000.,TIMER(R1) ;START 10 SEC TIMER AGAIN
8814 040640 006201          ASR      R1         ;
8815 040642 000433          BR       3$         ;EXIT
8816 040644 105761 034410          1$:     TSTB    DPINT(R1)  ;INITIALIZING THE DRIVE ?
8817 040650 001424          BEQ     2$         ;BR IF NOT
8818 040652 105061 034410     CLRB   DPINT(R1)  ;CLEAR THE INIT INDICATOR
8819 040656 004037 034746     JSR    RO,DRVINT  ;GO INIT THE DRIVE
8820 040662 000240          NOP          ;DUMMY PARITY ERROR RETURN
8821 040664 105761 034370     TSTB   DRVSTA(R1) ;DRIVE ONLINE ?
8822 040670 003014          BGT     2$         ;BR IF YES -- START ORDER
8823 040672 005702          TST     R2         ;QUEUE ENTRY FOR THE DRIVE
8824 040674 001416          BEQ     3$         ;BR IF NOT
8825 040676 004737 042672     JSR    PC,GETREQ  ;GET DPB ADDRESS
8826 040702 052762 140000 000016 BIS     #BIT15!BIT14,16(R2) ;INFORM USER THAT DRIVE OFFLINE
8827 040710 004737 042036     JSR    PC,SVRH11  ;SAVE THE REGISTERS
8828 040714 004737 042576     JSR    PC,EMPTYQ  ;EMPTY THE REQUEST QUEUE
8829 040720 000404          BR       3$         ;
8830 040722 105061 034420          2$:     CLRB   DPRQS(R1) ;CLEAR THE PORT REQUEST INDICATOR
8831 040726 004737 035576     JSR    PC,OPT     ;START THE PENDING REQUEST
8832 040732 000137 037660          3$:     JMP     SC4         ;PROCESS OTHER DRIVES
8833
8834          ;RPO4/5/6 TIMER ROUTINE
8835          ;CALL
8836          ;
8837          ;     MOV     #TIME,-(SP) ;ELAPSED TIME IN MILLISECONDS ON THE STACK
8838          ;     JSR    PC,RPTMR  ;CALL RPO4/5/6 TIME ROUTINE
8839 040736 005737 034434          RPTMR: TST     ACTDRV  ;CHECK "ACTDRV & ACTSTR"
8840 040742 001030          BNE     4$         ;IF NON ZERO EXIT
8841 040744 112737 000001 034435 MOVB    #1,ACTSTR  ;SET "ACTSTR"
8842 040752 104412          SAVREG ;SAVE R0 - R5
8843 040754 005001          CLR     R1         ;START WITH DRIVE 0
8844 040756 005003          CLR     R3         ;
8845 040760 005763 034462          1$:     TST     TIMER(R3) ;IS THE TIMER RUNNING?
8846 040764 002407          BLT    2$         ;BRANCH IF NO
8847 040766 166663 000002 034462 SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
8848 040774 003003          BGT    2$         ;BR IF NO SOFTWARE TIMEOUT
8849 040776 004737 041030     JSR    PC,STO     ;CALL SOFTWARE TIMEOUT ROUTINE
8850 041002 000405          BR     3$         ;GO TO THE EXIT
8851 041004 005201          2$:     INC     R1         ;MOVE TO NEXT DRIVE
8852 041006 005723          TST    (R3)+      ;
8853 041010 022701 000010     CMP    #8.,R1     ;OUT OF DRIVES?
8854 041014 003361          BGT    1$         ;BRANCH IF NO
8855 041016 104413          3$:     RESREG ;RESTORE R0 - R5
8856 041020 105037 034435     CLRB   ACTSTR    ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
8857 041024 012616          4$:     MOV    (SP)+,(SP) ;ADJUST THE STACK
  
```

```

8858 041026 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 041030 010146          STO:    MOV      R1,-(SP)      ;SAVE R1
8871 041032 010346          MOV      R3,-(SP)      ;SAVE R3
8872 041034 013704 034516  MOV      RPADR,R4      ;GET ADDRESS OF "RPCS1"
8873 041040 010164 000010  MOV      R1,RPCS2(R4)  ;SELECT THE DRIVE
8874 041044 004037 041464  JSR      R0,RD.RP      ;READ "DRIVE STATUS REG"
8875 041050 000012          RPDS1
8876 041052 041352          ST05
8877 041054 105726          TSTB    (SP)+          ;IS "DRY"=1?
8878 041056 100477          BMI     ST02          ;BR IF YES
8879 041060 105761 034410  ST01:  TSTB    DPINT(R1)    ;TRYING TO INTIALIZE THE DRIVE ?
8880 041064 001074          BNE     ST02          ;BR IF YES
8881 041066 105761 034420  TSTB    DPRQS(R1)    ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
8882 041072 001071          BNE     ST02          ;BR IF YES
8883 041074 013702 034430  MOV      TRNSWT,R2    ;PICKUP TRANSFER WAIT QUEUE
8884 041100 020137 034502  CMP     R1,DTUW      ;TRANSFER UNDERWAY ON THIS DRIVE?
8885 041104 001402          BEQ     1$           ;BRANCH IF YES
8886 041106 004737 042672  JSR      PC,GETREQ    ;GET DPB ADDRESS
8887 041112 052762 101000 000016 1$:    BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
8888 041120 004737 042036          JSR      PC,SVRH11    ;SAVE RH11/RPO4/5/6 REGISTERS
8889 041124 012764 000040 000010  MOV     #BIT05,RPCS2(R4) ;"INIT" THE MASS BUS
8890 041132 105061 034360  CLRB   DRVACT(R1)    ;DRIVE IS IDLE
8891 041136 105061 034436  CLRB   ULDFLG(R1)    ;CLEAR THE UNLOAD FLAG
8892 041142 005001          CLR     R1           ;START WITH DRIVE 0
8893 041144 005003          CLR     R3
8894 041146 004037 034746  2$:    JSR      R0,DRVINT    ;INIT. THIS DRIVE
8895 041152 000477          BR      ST05          ;PARITY ERROR RETURN
8896 041154 105761 034360  TSTB   DRVACT(R1)    ;DRIVE IDLE BEFORE THE INIT. ?
8897 041160 001414          BEQ     4$           ;YES--BRANCH
8898 041162 013702 034430  MOV     TRNSWT,R2    ;GET TRANSFER WAIT QUEUE
8899 041166 023701 034502  CMP     DTUW,R1      ;WAS THERE I/O ON THIS DRIVE?
8900 041172 001402          BEQ     3$           ;YES--BRANCH
8901 041174 004737 042672  JSR     PC,GETREQ    ;GET THE DPB POINTER FROM QUEUE
8902 041200 052762 100400 000016 3$:    BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
8903 041206 105061 034360  CLRB   DRVACT(R1)    ;SET DRIVE ACTIVE TO IDLE
8904 041212 105061 034436  4$:    CLRB   ULDFLG(R1)    ;NO UNLOAD
8905 041216 012763 177777 034462  MOV     #-1,TIMER(R3) ;STOP THE TIMER
8906 041224 005723          TST    (R3)+         ;UPDATE THE INDEX
8907 041226 005201          INC    R1           ;INCREMENT THE DRIVE NUMBER
8908 041230 022701 000010  CMP     #8.,R1      ;LAST DRIVE BEEN CHECKED?
8909 041234 003344          BGT    2$           ;NO--LOOP
8910 041236 012737 177777 034502  MOV     #-1,DTUW     ;NO DATA TRANSFERS UNDERWAY
8911 041244 005037 034430  CLR     TRNSWT      ;CLEAR TRANSFER WAIT QUEUE
8912 041250 004737 042520  JSR     PC,CLRQUE    ;CLEAR ALL REQUEST QUEUES
8913 041254 000500          BR     ST09          ;EXIT

```



```

8914 041256 116405 000016          ST02:  MOVB   RPAS(R4),R5      ;READ ATTENTION REG
8915 041262 136105 034504          BITB   ATABIT(R1),R5      ;IS ATTENTION FOR THIS DRIVE ?
8916 041266 001017                   BNE    ST03              ;YES--BRANCH
8917 041270 105761 034410          TSTB  DPINT(R1)         ;TRYING TO INITIALIZE THE DRIVE ?
8918 041274 001031                   BNE    ST06              ;BR IF YES - DRIVE NOT ONLINE
8919 041276 105761 034420          TSTB  DPRQS(R1)        ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
8920 041302 001045                   BNE    ST07              ;BR IF YES - NO RESPONSE TO REQUEST
8921 041304 020137 034502          CMP   R1,DTUW          ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
8922 041310 001263                   BNE    ST01              ;BR IF NO
8923 041312 004037 041464          JSR   RO,RD.RP         ;YES--CHECK "RDY"
8924 041316 000000                   RPCS1
8925 041320 041352                   ST05
8926 041322 105726                   TSTB  (SP)+
8927 041324 100255                   BPL   ST01              ;BR IF "RDY"=0
8928 041326 105761 034410          ST03:  TSTB  DPINT(R1)         ;INITIALIZING THE DRIVE ?
8929 041332 001003                   BNE    15                ;BR IF INIT PENDING
8930 041334 105761 034420          TSTB  DPRQS(R1)        ;PORT REQUEST PENDING ?
8931 041340 001446                   BEQ   ST09              ;BR IF NOT
8932 041342 012763 177777 034462  15:   MOV   #-1,TIMER(R3)     ;STOP THE TIMER
8933 041350 000442                   BR    ST09              ;EXIT
8934 041352 004737 037006          ST05:  JSR   PC,C18        ;GO HANDLE THE PARITY ERROR
8935 041356 000437                   BR    ST09
8936 041360 105061 034410          ST06:  CLRB  DPINT(R1)         ;CLEAR THE INITIALIZE INDICATOR
8937 041364 105061 034370          CLRB  DRVSTA(R1)       ;SET UNIT OFFLINE
8938 041370 012763 177777 034462  MOV   #-1,TIMER(R3)     ;STOP THE TIMER
8939 041376 004737 042672          JSR   PC,GETREQ        ;GET THE DPB ADDRESS
8940 041402 005702                   TST   R2                ;REQUEST IN QUEUE ?
8941 041404 001424                   BEQ   ST09              ;BR IF NOT
8942 041406 052762 140000 000016  BIS   #BIT15!BIT14,16(R2) ;INFORM THE USER DRIVE NOT AVAILABLE
8943 041414 000414                   BR    ST08              ;FINISH
8944 041416 012763 177777 034462  ST07:  MOV   #-1,TIMER(R3)     ;STOP THE TIMER
8945 041424 105061 034420          CLRB  DPRQS(R1)        ;CLEAR PORT REQUEST INDICATOR
8946 041430 004737 042672          JSR   PC,GETREQ        ;GET DPB ADDRESS
8947 041434 005702                   TST   R2                ;QUEUE ENTRY FOR DRIVE ?
8948 041436 001407                   BEQ   ST09              ;BR IF NONE
8949 041440 012762 100004 000016  MOV   #BIT15!BIT2,16(R2) ;INFORM USER OF PORT REQUEST ERROR
8950 041446 004737 042576          ST08:  JSR   PC,EMPTYQ      ;CLEAR THE QUEUE FOR THE DRIVE
8951 041452 004737 042036          JSR   PC,SVRH11        ;SAVE THE REGISTERS
8952 041456 012603          ST09:  MOV   (SP)+,R3      ;RESTORE R3
8953 041460 012601          MOV   (SP)+,R1        ;RESTORE R1
8954 041462 000207          RTS   PC                ;RETURN
8955
8956          ;ROUTINE TO READ A RH11/RPO4/5/6 REGISTER
8957          ;
8958          ;CALL
8959          ;   JSR   RO,RD.RP      ;GO READ A REGISTER
8960          ;   INDEX FROM BASE
8961          ;   ERRADR          ;ERROR ADDRESS--PROCESS ERROR STARTING
8962          ;   ;AT THIS ADDRESS
8963          ;   ;CONTENTS OF REG. IS ON THE STACK
8964          ;
8965 041464 013737 034514 041632  RD.RP:  MOV   MCPMX,RD.RP2    ;MAX. RETRYS ALLOWED
8966 041472 011646                   MOV   (SP),-(SP)        ;SAVE RO FOR RETURN
8967 041474 013737 034516 041510  MOV   RPADR,RD.ADR     ;FORM THE DESIRED ADDRESS
8968 041502 062037 041510          ADD   (RO)+,RD.ADR     ;USING THE BASE AND THE INDEX
8969 041506 013727          RD.RP1: MOV   @((PC)+,(PC)+ ;READ THE DESIRED REGISTER OF THE RPO4
    
```

```

8970 041510 000000 RD. ADR: . WORD 0 ; ADDRESS IS FORMED HERE
8971 041512 000000 RD. WRD: . WORD 0 ; REG. CONTENTS PUT HERE
8972 041514 013766 041512 000002 MOV RD. WRD, 2(SP) ; RETURN IT TO THE USER
8973 041522 013746 034516 MOV RPADR, -(SP) ; PUT THE ADDRESS ON THE STACK
8974 041526 062716 000010 ADD @RPCS2, (SP) ; FORM THE ADDRESS OF RPCS2
8975 041532 032736 010000 BIT @BIT12, @ (SP)+ ; CHECK THE 'MED' BIT
8976 041536 001037 BNE RD. RP3 ; BR IF DRIVE NON-EXISTENT
8977 041540 017746 172752 MOV @RPADR, -(SP) ; READ RPCS1
8978 041544 032716 020000 BIT @BIT13, (SP) ; DID MCPE SET?
8979 041550 001002 BNE 1$ ; BRANCH IF YES
8980 041552 022620 CMP (SP)+, (RO)+ ; ADJUST FOR RETURN
8981 041554 000432 BR RD. RP4 ; EXIT
8982 041556
8983 041556 004037 042756 15: JSR RO, ES. SAV ; SAVE THE ADDRESS IN 'SESCAPE'
8984 041562 104003 ERROR 3 ; REPORT "MCPE" ERROR
8985 041564 005737 034502 TST DTUW ; DATA TRANSFER UNDERWAY?
8986 041570 100405 BMI 2$ ; NO--BRANCH
8987 041572 032716 040000 BIT @BIT14, (SP) ; NO--"TRE"=1?
8988 041576 001402 BEQ 2$ ; NO--BRANCH
8989 041600 005726 TST (SP)+ ; YES--CLEAN OFF THE STACK AND
8990 041602 000415 BR RD. RP3 ; TAKE THE FATAL ERROR EXIT
8991 041604 052716 040000 25: BIS @BIT14, (SP) ; CLEAR "MCPE" BY SENDING A "1" TO "TRE"
8992 041610 000316 SWAB (SP) ; POSITION BEFORE WRITING
8993 041612 013737 034516 041626 MOV RPADR, 3$ ; FORM ADDRESS OF HIGH BYTE
8994 041620 005237 041626 INC 3$
8995 041624 112637 MOVVB (SP)+, @ (PC)+ ; WRITE THE HIGH BYTE OF RPCS1
8996 041626 000000 35: . WORD 0 ; ADDRESS STORAGE
8997 041630 005327 DEC (PC)+ ; EXCEEDED MAX. RETRYS
8998 041632 000003 RD. RP2: . WORD 3
8999 041634 002324 BGE RD. RP1 ; BRANCH IF NO
9000 041636 011000 RD. RP3: MOV (RO), RO ; FATAL ERROR EXIT
9001 041640 012616 RD. RP4: MOV (SP)+, (SP)
9002 041642 000200 RTS RO
9003
9004 ; ROUTINE TO WRITE A REGISTER
9005 ;
9006 ; CALL
9007 ;
9008 ; MOV DATA, -(SP) ; DATA TO BE LOADED ON THE STACK
9009 ; JSR RO, WRT. RP ; CALL THE ROUTINE TO LOAD(WRITE) THE REG.
9010 ; INDEX ; INDEX OF THE REGISTER TO BE LOADED
9011 ; ERRADR ; ADDRESS TO RETURN TO ON AN ERROR
9012 ; RETURN ; ERROR FREE RETURN
9013 041644 013737 034514 042020 WRT. RP: MOV MCPENX, WRT. R2 ; MAX RETRYS ALLOWED
9014 041652 016637 000002 041732 MOV 2(SP), WRT. WD ; SAVE THE WORD TO WRITE
9015 041660 012616 MOV (SP)+, (SP) ; ADJUST THE STACK
9016 041662 012037 041734 MOV (RO)+, WRT. AD ; GET INDEX OF REGISTER TO BE WRITTEN
9017 041666 001015 BNE 1$ ; BRANCH IF NOT RPCS1
9018 041670 122737 000150 041732 CMPB #150, WRT. WD ; IS THE COMMAND FOR DATA TRANSFERS?
9019 041676 002411 BLT 1$ ; YES--DON'T GET THE OLD A16 & A17, & PSEL
9020 041700 004037 041464 JSR RO, RD. RP ; NO---COMBINE A16&A17, & PSEL WITH
9021 041704 000000 RPCS1 ; THE COMMAND BEFORE SENDING IT TO
9022 041706 042026 WRT. R3 ; THE RH11/RPO4
9023 041710 000316 SWAB (SP)
9024 041712 042716 177770 BIC # C7, (SP)
9025 041716 112637 041733 MOVVB (SP)+, WRT. WD+1
    
```



```

9026 041722 063737 034516 041734 15: ADD RPADR, WRT. AD ; FORM THE ADDRESS OF THE DISK REG.
9027 041730 012737 WRT. R1: MOV (PC)+, @ (PC)+ ; LOAD THE DESIRED REG.
9028 041732 000000 WRT. WD: . WORD 0 ; WORD TO WRITE GOES HERE
9029 041734 000000 WRT. AD: . WORD 0 ; ADDRESS IS FORMED HERE
9030 041736 013746 034516 MOV RPADR, -(SP) ; PUT THE ADDRESS ON THE STACK
9031 041742 062716 000010 ADD @RPCS2, (SP) ; FORM THE ADDRESS OF RPCS2
9032 041746 032736 010000 BIT #BIT12, @ (SP)+ ; CHECK THE 'MED' BIT
9033 041752 001025 BNE WRT. R3 ; BR IF DRIVE NON-EXISTENT
9034 041754 004037 041464 JSR RO, RD. RP ; CHECK FOR PARITY ERROR ON WRITE
9035 041760 000014 RPER1
9036 041762 042026 WRT. R3
9037 041764 032726 000010 BIT #BIT03, (SP)+
9038 041770 001420 BEQ WRT. R4 ; BRANCH IF "PAR=0"
9039 041772 016037 177776 042004 MOV -2(RO), 15 ; PICKUP THE INDEX
9040 042000 004037 041464 JSR RO, RD. RP ; READ THE REG.
9041 042004 000000 15: . WORD 0 ; REG. INDEX
9042 042006 042026 WRT. R3 ; RETURN TO THIS ADDRESS ON ERROR
9043 042010 004037 042756 JSR RO, ES. SAV ; SAVE THE ADDRESS IN 'SESCAPE'
9044 042014 104004 ERROR 4 ; REPORT THE PARITY ON WRITE ERROR
9045 042016 005327 DEC (PC)+ ; DECREMENT THE ERROR COUNT
9046 042020 000003 WRT. R2: . WORD 3 ; RETRY COUNTER
9047 042022 002342 BGE WRT. R1 ; TRY AGAIN IF NOT FINISHED
9048 042024 005726 TST (SP)+ ; CLEAN OFF THE STACK
9049 042026 011000 WRT. R3: MOV (RO), RO ; TAKE THE "PARITY ON WRITE" ERROR EXIT
9050 042030 000401 BR WRT. R5 ; EXIT
9051 042032 005720 WRT. R4: TST (RO)+ ; ADJUST FOR ERROR FREE EXIT
9052 042034 000200 WRT. R5: RTS RO
9053
9054 ; ROUTINE TO SAVE THE RH11/RPO4/5/6 REGISTERS AS PER DPB+14
9055 ;
9056 ; CALL
9057 ;
9058 ; MOV #DPBNUM, R2 ; DPB POINTER TO R2
9059 ; JSR PC, SVRH11 ; SAVE THE DRIVES REG'S
9060 042036 104412 SVRH11: SAVREG ; SAVE RO - R5
9061 042040 005702 TST R2 ; QUEUE ENTRY FOR THE DRIVE ?
9062 042042 001430 BEQ 45 ; BR IF NONE
9063 042044 013704 034516 MOV RPADR, R4
9064 042050 111264 000010 MOVB (R2), RPCS2(R4) ; SELECT DRIVE
9065 042054 016203 000014 MOV 14(R2), R3 ; GET THE ERROR TABLE POINTER
9066 042060 001433 BEQ 65 ; EXIT IF NO ADDRESS
9067 042062 005037 042116 CLR 35 ; COUNTER & POINTER
9068 042066 023727 042116 000022 15: CMP 35, #RPDB ; REACHED THE BUFFER REGISTER ?
9069 042074 001006 BNE 25 ; BR IF NOT
9070 042076 032764 000200 000010 BIT #BIT07, RPCS2(R4) ; 'OR' SET ?
9071 042104 001002 BNE 25 ; BR IF SET
9072 042106 005023 CLR (R3)+ ; STORE RPDB AS ZEROES
9073 042110 000405 BR 45 ; CONTINUE
9074 042112 004037 041464 25: JSR RO, RD. RP ; READ THE SELECTED REGISTER
9075 042116 000000 35: . WORD 0 ; REGISTER INDEX
9076 042120 042144 55 ; ERROR RETURN ADDRESS
9077 042122 012623 MOV (SP)+, (R3)+ ; STORE THE REGISTER CONTENTS
9078 042124 023727 042116 000046 45: CMP 35, #RPEC2 ; REACHED THE END ?
9079 042132 001406 BEQ 65 ; BR IF YES
9080 042134 062737 000002 042116 ADD #2, 35 ; INCREMENT THE REGISTER INDEX
9081 042142 000751 BR 15 ; CONTINUE READING THE REGISTERS
    
```

```

9082 042144 004737 036706      55:   JSR    PC,C17      ;PROCESS THE UNCORRECTABLE PARITY ERROR
9083 042150 104413              65:   RESREG                ;RESTORE R0 - R5
9084 042152 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 042154 010446              SET. IE: MOV    R4,-(SP)      ;SAVE R4
9093 042156 013704 034516      MOV    RPADR,R4        ;PICKUP ADDRESS OF RPCS1
9094 042162 010164 000010      MOV    R1,RPCS2(R4)   ;SELECT DRIVE
9095 042166 011446              MOV    (R4),-(SP)     ;READ RPCS1
9096 042170 052716 040000      BIS    #BIT14,(SP)    ;SET THE "TRE" BIT OF THE WORD READ
9097 042174 000316              SWAB  (SP)            ;ADJUST FOR DATO
9098 042176 112714 000100      MOVB  #BIT06,(R4)     ;SET "IE"
9099 042202 032764 010000 000010 BIT    #BIT12,RPCS2(R4) ;IS "NED"=1?
9100 042210 001002              BNE    1$             ;YES--CLEAR "TRE"
9101 042212 005726              TST   (SP)+           ;CLEAN OFF THE STACK
9102 042214 000402              BR    2$
9103 042216 112664 000001      1$:  MOVB  (SP)+,1(R4)    ;CLEAR "TRE"
9104 042222 012604              2$:  MOV   (SP)+,R4      ;RESTORE R4
9105 042224 000207              RTS    PC                ;RETURN TO CALLER
9106
9107      ;QUEUE COUNT
9108 042226      000      QCNT:  .BYTE  0            ;DRIVE 0
9109 042227      000      .BYTE  0            ;DRIVE 1
9110 042230      000      .BYTE  0            ;DRIVE 2
9111 042231      000      .BYTE  0            ;DRIVE 3
9112 042232      000      .BYTE  0            ;DRIVE 4
9113 042233      000      .BYTE  0            ;DRIVE 5
9114 042234      000      .BYTE  0            ;DRIVE 6
9115 042235      000      .BYTE  0            ;DRIVE 7
9116
9117      ;QUEUE INPUT POINTERS
9118
9119 042236 042320      QINPT: .WORD  QDRV0        ;DRIVE 0
9120 042240 042340      .WORD  QDRV1        ;DRIVE 1
9121 042242 042360      .WORD  QDRV2        ;DRIVE 2
9122 042244 042400      .WORD  QDRV3        ;DRIVE 3
9123 042246 042420      .WORD  QDRV4        ;DRIVE 4
9124 042250 042440      .WORD  QDRV5        ;DRIVE 5
9125 042252 042460      .WORD  QDRV6        ;DRIVE 6
9126 042254 042500      .WORD  QDRV7        ;DRIVE 7
9127
9128      ;QUEUE OUTPUT POINTERS
9129
9130 042256 042320      QOUTPT: .WORD  QDRV0        ;DRIVE 0
9131 042260 042340      .WORD  QDRV1        ;DRIVE 1
9132 042262 042360      .WORD  QDRV2        ;DRIVE 2
9133 042264 042400      .WORD  QDRV3        ;DRIVE 3
9134 042266 042420      .WORD  QDRV4        ;DRIVE 4
9135 042270 042440      .WORD  QDRV5        ;DRIVE 5
9136 042272 042460      .WORD  QDRV6        ;DRIVE 6
9137 042274 042500      .WORD  QDRV7        ;DRIVE 7

```



```

9138
9139 042276 042320          QSTART: . WORD   QDRV0          ; DRIVE 0 START ADDRESS
9140 042300 042340          QSTOP:  . WORD   QDRV1          ; DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
9141 042302 042360          . WORD   QDRV2          ; STOP DRIVE 1--START DRIVE 2
9142 042304 042400          . WORD   QDRV3          ; STOP DRIVE 2--START DRIVE 3
9143 042306 042420          . WORD   QDRV4          ; STOP DRIVE 3--START DRIVE 4
9144 042310 042440          . WORD   QDRV5          ; STOP DRIVE 4--START DRIVE 5
9145 042312 042460          . WORD   QDRV6          ; STOP DRIVE 5--START DRIVE 6
9146 042314 042500          . WORD   QDRV7          ; STOP DRIVE 6--START DRIVE 7
9147 042316 042520          . WORD   QTERM          ; STOP DRIVE 7
9148
9149          ; DRIVE REQUEST QUEUES
9150
9151 042320 000010          QDRV0: . BLKW   10
9152 042340 000010          QDRV1: . BLKW   10
9153 042360 000010          QDRV2: . BLKW   10
9154 042400 000010          QDRV3: . BLKW   10
9155 042420 000010          QDRV4: . BLKW   10
9156 042440 000010          QDRV5: . BLKW   10
9157 042460 000010          QDRV6: . BLKW   10
9158 042500 000010          QDRV7: . BLKW   10
9159          QTERM=.
9160
9161          ; ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
9162
9163          ; CALL
9164          ;     JSR     PC, CLRQUE
9165
9166 042520 104412          CLRQUE: SAVREG          ; SAVE R0 - R5
9167 042522 012702 042226  MOV     #QCNT, R2      ; ZERO THE QUEUE COUNTS
9168 042526 005022          CLR     (R2)+          ; DRIVES 0 & 1
9169 042530 005022          CLR     (R2)+          ; DRIVES 2 & 3
9170 042532 005022          CLR     (R2)+          ; DRIVES 4 & 5
9171 042534 005022          CLR     (R2)+          ; DRIVES 6 & 7
9172 042536 012703 000010  MOV     #8, R3         ; MOVE THE STARTING
9173 042542 012701 042276  MOV     #QSTART, R1    ; ADDRESS OF THE QUEUE INTO
9174 042546 012122          15:   MOV     (R1)+, (R2)+ ; THE QUEUE INPUT POINTER
9175 042550 005303          DEC     R3
9176 042552 001375          BNE     15
9177 042554 012703 000010  MOV     #8, R3         ; MOVE THE STARTING ADDRESS
9178 042560 012701 042276  MOV     #QSTART, R1    ; OF THE QUEUE INTO THE
9179 042564 012122          25:   MOV     (R1)+, (R2)+ ; QUEUE OUTPUT POINTER
9180 042566 005303          DEC     R3
9181 042570 001375          BNE     25
9182 042572 104413          RESREG
9183 042574 000207          RTS     PC             ; RESTORE R0 - R5
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 042576 105061 042226  EMPTYQ: CLRB   QCNT(R1) ; CLEAR NUMBER OF ITEMS IN QUEUE
9192 042602 006301          ASL     R1
9193 042604 016161 042236 042256  MOV     QINPT(R1), QOUTPT(R1) ; SET OUTPUT QUEUE POINTER=INPUT POINTER
    
```

```

9194 042612 006201          ASR      R1
9195 042614 000207          RTS      PC
9196
9197          ;ROUTINE TO PUT A REQUEST IN QUEUE
9198          ;
9199          ;CALL
9200          ;
9201          ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER
9202          ;      MOV      #DPB,R2        ;ADDRESS OF PARAMETER BLOCK
9203          ;      JSR      RO,DRVQUE     ;GO PUT REQUEST IN QUEUE
9204          ;      RETURN1   ;RETURN HERE IF QUEUE IS FULL
9205          ;      RETURN2   ;RETURN HERE IF REQUEST IS IN QUEUE
9206 042616 122761 000010 042226 DRVQUE: CMPB    #10,QCNT(R1) ;IS QUEUE FULL?
9207 042624 001421          BEQ     25          ;BR IF YES-TAKE RETURN1
9208 042626 105261 042226          INCB   QCNT(R1)   ;INCREMENT QUEUE COUNT
9209 042632 006301          ASL    R1
9210 042634 010271 042236          MOV    R2,QINPT(R1) ;PUT THIS REQUEST IN QUEUE
9211 042640 062761 000002 042236          ADD   #2,QINPT(R1) ;UPDATE THE QUEUE POINTER
9212 042646 026161 042236 042300          CMP   QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
9213 042654 001003          BNE   15          ;BRANCH IF NO
9214 042656 016161 042276 042236          MOV   QSTART(R1),QINPT(R1) ;YES--RESET POINTER
9215 042664 006201          15:   ASR    R1
9216 042666 005720          TST   (R0)+      ;TAKE RETURN 2
9217 042670 000200          25:   RTS    RO      ;RETURN TO USER
9218
9219          ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
9220          ;
9221          ;CALL
9222          ;
9223          ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
9224          ;      JSR      PC,GETREQ     ;GO GET THE REQUEST
9225          ;      RETURN   ;R2="DPB" ADDRESS OF THE REQUEST
9226          ;      ;R2=0 IF NO REQUEST IN QUEUE
9227 042672 005002          GETREQ: CLR    R2
9228 042674 105761 042226          TSTB  QCNT(R1)   ;IS THERE ANY REQUEST IN QUEUE?
9229 042700 001404          BEQ   25          ;NO---BRANCH
9230 042702 006301          15:   ASL    R1
9231 042704 017102 042256          MOV   @QOUTPT(R1),R2 ;PICKUP "DPB" POINTER FOR THIS DRIVE
9232 042710 006201          ASR   R1
9233 042712 000207          25:   RTS    PC      ;RETURN TO USER
9234
9235          ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
9236          ;
9237          ;CALL
9238          ;
9239          ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
9240          ;      JSR      PC,POPQUE     ;CALL TO REMOVE REQUEST
9241          ;      RETURN   ;R2=ADDRESS OF DPB REMOVED
9242 042714 105361 042226          POPQUE: DECB   QCNT(R1) ;DECREMENT QUEUE COUNT
9243 042720 006301          ASL   R1
9244 042722 017102 042256          MOV   @QOUTPT(R1),R2 ;GET THE "DPB" POINTER
9245 042726 062761 000002 042256          ADD   #2,QOUTPT(R1) ;UPDATE THE QUEUE POINTER
9246 042734 026161 042256 042300          CMP   QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
9247 042742 001003          BNE   15          ;NO--BRANCH TO EXIT
9248 042744 016161 042276 042256          MOV   QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
9249 042752 006201          15:   ASR    R1
    
```



```

9250 042754 000207          RTS      PC          ;RETURN TO USER
9251
9252          ;ROUTINE TO SAVE THE CONTENTS OF 'SESCAPE' WHEN THE DRIVER
9253          ;REPORTS AN ERROR DIRECTLY.
9254          ;
9255          ;CALL
9256          ;
9257          ;      JSR      RO,ES.SAV
9258          ;      ERROR  N          ;; THE ERROR CALL
9259          ;      RETURN          ;THE RETURN IS PAST THE ERROR CALL
9260 042756 012037 042772    ES.SAV: MOV      (RO)+,1$          ;GET THE ERROR CALL
9261 042762 013746 001206    MOV      $ESCAPE,-(SP)        ;SAVE THE ADDRESS IN 'SESCAPE'
9262 042766 005037 001206    CLR      $ESCAPE             ;CLEAR THE ESCAPE RETURN
9263 042772 000000          1$:      .WORD  0             ;THE ERROR CALL IS MOVED HERE
9264 042774 012637 001206    MOV      (SP)+,$ESCAPE        ;RESTORE THE ESCAPE ADDRESS
9265 043000 000200          RTS      RO          ;RETURN
9266
9267
9268          .SBTTL  ASCIZ MESSAGES
9269
9270 043002 000122          MSG.R:  .ASCIZ  /R/
9271 043004 041506          MSG.FC:  .ASCIZ  /FC/
9272 043007          114 000103    MSG.LC:  .ASCIZ  /LC/
9273 043012 041506 000047    MSGFCP: .ASCIZ  /FC'/
9274 043016 041514 000047    MSGLCP: .ASCIZ  /LC'/
9275 043022 041511          000    MSG.IC:  .ASCIZ  /IC/
9276 043025          106 000124    MSG.FT: .ASCIZ  /FT/
9277 043030 052114          000    MSG.LT: .ASCIZ  /LT/
9278 043033          111 000124    MSG.IT: .ASCIZ  /IT/
9279 043036 051506          000    MSG.FS: .ASCIZ  /FS/
9280 043041          114 000123    MSG.LS: .ASCIZ  /LS/
9281 043044 040520 000124    MSG.PAT: .ASCIZ  /PAT/
9282 043050 000075          MSG.EQ: .ASCIZ  /=/
9283 043052 005015 047503 052116    MSG.CS: .ASCIZ  <CR><LF>/CONTROL SWITCHES=/
9284 043060 047522 020114 053523
9285 043066 052111 044103 051505
9286 043074 000075
9287
9288 043076 027440 000040          SLASH:  .ASCIZ  @ / @
9289 043102 047125 052111 051440    SYSTAT: .ASCIZ  /UNIT STATUS: /<CR><LF><LF>
9290 043110 040524 052524 035123
9291 043116 005015 000012
9292 043122 051104 053111 000105    UNTMSG: .ASCIZ  /DRIVE/
9293 043130 047440 043106 044514    UNTOFF: .ASCIZ  / OFFLINE/
9294 043136 042516          000
9295 043141          040 047117 044514    UNTON:  .ASCIZ  / ONLINE/
9296 043146 042516          000
9297 043151          040 047516 020124    NOTPRS: .ASCIZ  / NOT PRESENT/
9298 043156 051120 051505 047105
9299 043164 000124
9300 043166 052440 051516 043101    NOTSAF: .ASCIZ  / UNSAFE/
9301 043174 000105
9302 043176 047040 052117 051040    NOTRP:  .ASCIZ  @ NOT RPO4/5/6@
9303 043204 030120 027464 027465
9304 043212 000066
9305 043214 050122 032060          000 RPO4B: .ASCIZ  /RPO4/
    
```

9306	043221	122	030120	000065	RPO5: .ASCIZ /RPO5/
9307	043226	050122	033060	000	RPO6: .ASCIZ /RPO6/
9308	043233	015	042012	044522	DRIVES: .ASCIZ <CR><LF>/DRIVE(S) TO BE TESTED /
9309	043240	042526	051450	020051	
9310	043246	047524	041040	020105	
9311	043254	042524	052123	042105	
9312	043262	000040			
9313	043264	047516	042516	000	NONE: .ASCIZ /NONE/
9314	043271	054	000		COMMA: .ASCIZ /,/
9315	043273	015	047012	020117	NOCLOCK: .ASCIZ <CR><LF>/NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
9316	043300	053513	030461	050055	
9317	043306	041440	047514	045503	
9318	043314	020054	044524	044515	
9319	043322	043516	052040	051505	
9320	043330	051524	053440	046111	
9321	043336	020114	047516	020124	
9322	043344	042502	050040	051105	
9323	043352	047506	046522	042105	
9324	043360	000			
9325	043361	015	005012	042524	TESTNG: .ASCIZ <CR><LF><LF>/TESTING DRIVE /
9326	043366	052123	047111	020107	
9327	043374	051104	053111	020105	
9328	043402	000			
9329	043403	123	051105	040511	SERIAL: .ASCIZ /SERIAL NUMBER /
9330	043410	020114	052516	041115	
9331	043416	051105	000040		
9332					
9333	043422	005015	051012	052117	MSG7: .ASCIZ <CR><LF><LF>/ROTATIONAL SPEED TIMES/
9334	043430	052101	047511	040516	
9335	043436	020114	050123	042505	
9336	043444	020104	044524	042515	
9337	043452	000123			
9338	043454	005015	047412	042516	MSG10A: .ASCIZ <CR><LF><LF>/ONE CYLINDER SEEK TIMES/<CR><LF>/ * FORWARD/
9339	043462	041440	046131	047111	
9340	043470	042504	020122	042523	
9341	043476	045505	052040	046511	
9342	043504	051505	005015	025040	
9343	043512	043040	051117	040527	
9344	043520	042122	000		
9345	043523	015	020012	020052	MSG10B: .ASCIZ <CR><LF>/ * REVERSE/
9346	043530	042522	042526	051522	
9347	043536	000105			
9348	043540	005015	040412	041503	MSG11A: .ASCIZ <CR><LF><LF>/ACCESS TIME MEASUREMENT/<CR><LF>/ * FORWARD/
9349	043546	051505	020123	044524	
9350	043554	042515	046440	040505	
9351	043562	052523	046522	047105	
9352	043570	006524	020012	020052	
9353	043576	047506	053522	051101	
9354	043604	000104			
9355	043606	005015	025040	051040	MSG11B: .ASCIZ <CR><LF>/ * REVERSE/
9356	043614	053105	051105	042523	
9357	043622	000			
9358	043623	015	005012	040515	MSG12A: .ASCIZ <CR><LF><LF>/MAXIMUM SEEK TIMES/<CR><LF>/ * FORWARD/
9359	043630	044530	052515	020115	
9360	043636	042523	045505	052040	
9361	043644	046511	051505	005015	



9362	043652	025040	043040	051117	
9363	043660	040527	042122	000	
9364	043665	015	020012	020052	MSG12B: .ASCIZ <CR><LF>/ * REVERSE/
9365	043672	042522	042526	051522	
9366	043700	000105			
9367					
9368	043702	005015	044515	036516	MSGMIN: .ASCIZ <CR><LF>/MIN=/ /
9369	043710	000			
9370	043711	015	046412	054101	MSGMAX: .ASCIZ <CR><LF>/MAX=/ /
9371	043716	000075			
9372	043720	005015	053101	036507	MSGAVG: .ASCIZ <CR><LF>/AVG=/ /
9373	043726	000			
9374	043727	060	052440	000123	MSGOUS: .ASCIZ /O US/ /
9375	043734	041040	046105	053517	MBELOW: .ASCIZ / BELOW THE MINIMUM OF / /
9376	043742	052040	042510	046440	
9377	043750	047111	046511	046525	
9378	043756	047440	020106	000	
9379	043763	040	041101	053117	MABOVE: .ASCIZ / ABOVE THE MAXIMUM OF / /
9380	043770	020105	044124	020105	
9381	043776	040515	044530	052515	
9382	044004	020115	043117	000040	
9383	044012	051440	042505	051513	MSGNUM: .ASCIZ / SEEKS TIMED/ /
9384	044020	052040	046511	042105	
9385	044026	000			
9386	044027	040	047516	020124	MSGNON: .ASCIZ / NOT TIMED/ /
9387	044034	044524	042515	000104	
9388	044042	020040	000		MSG. SP: .ASCIZ / / ; TWO (2) SPACES
9389					
9390					.SBTTL ERROR HEADER (EM) MESSAGES
9391					
9392	044045	122	030510	020061	EM1: .ASCIZ /RH11 INTERRUPT OCCURRED (RPAS = 0)/ /
9393	044052	047111	042524	051122	
9394	044060	050125	020124	041517	
9395	044066	052503	051122	042105	
9396	044074	024040	050122	051501	
9397	044102	036440	030040	000051	
9398	044110	047125	054105	042520	EM2: .ASCIZ /UNEXPECTED ATTENTION OCCURRED/ /
9399	044116	052103	042105	040440	
9400	044124	052124	047105	044524	
9401	044132	047117	047440	041503	
9402	044140	051125	042522	000104	
9403	044146	040515	051523	052502	EM3: .ASCIZ /MASSBUS PARITY ERROR(MCPE=1)/ /
9404	044154	020123	040520	044522	
9405	044162	054524	042440	051122	
9406	044170	051117	046450	050103	
9407	044176	036505	024461	000	
9408	044203	115	051501	041123	EM4: .ASCIZ /MASSBUS PARITY ERROR(PAR=1)/ /
9409	044210	051525	050040	051101	
9410	044216	052111	020131	051105	
9411	044224	047522	024122	040520	
9412	044232	036522	024461	000	
9413	044237	101	042104	042522	EM5: .ASCIZ /ADDRESS PLUG CHANGE BIT SET/ /
9414	044244	051523	050040	052514	
9415	044252	020107	044103	047101	
9416	044260	042507	041040	052111	
9417	044266	051440	052105	000	

9418	044273	122	030510	027461	EM10:	. ASCIZ "RH11/RPO4/5/6 FAILED TO RESPOND TO ADDRESSING"
9419	044300	050122	032060	032457		
9420	044306	033057	043040	044501		
9421	044314	042514	020104	047524		
9422	044322	051040	051505	047520		
9423	044330	042116	052040	020117		
9424	044336	042101	051104	051505		
9425	044344	044523	043516	000		
9426	044351	104	044522	042526	EM11:	. ASCIZ /DRIVE SELECTED IS NOT ONLINE/
9427	044356	051440	046105	041505		
9428	044364	042524	020104	051511		
9429	044372	047040	052117	047440		
9430	044400	046116	047111	000105		
9431	044406	046511	051120	050117	EM12:	. ASCIZ /IMPROPER HEADER DATA/
9432	044414	051105	044040	040505		
9433	044422	042504	020122	040504		
9434	044430	040524	000			
9435	044433	104	052101	020101	EM13:	. ASCIZ /DATA COMPARE FAILURE/
9436	044440	047503	050115	051101		
9437	044446	020105	040506	046111		
9438	044454	051125	000105			
9439	044460	044504	045523	042440	EM17:	. ASCIZ /DISK ERROR IN TIMING TEST/
9440	044466	051122	051117	044440		
9441	044474	020116	044524	044515		
9442	044502	043516	052040	051505		
9443	044510	000124				
9444	044512	046103	041517	020113	EM20:	. ASCIZ /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
9445	044520	045450	030527	026461		
9446	044526	024520	047440	042526		
9447	044534	043122	047514	020127		
9448	044542	047111	052040	046511		
9449	044550	047111	020107	042524		
9450	044556	052123	000			
9451	044561	104	051511	020113	EM23:	. ASCIZ /DISK ERROR DURING SEEK/
9452	044566	051105	047522	020122		
9453	044574	052504	044522	043516		
9454	044602	051440	042505	000113		
9455	044610	042523	045505	047040	EM24:	. ASCIZ /SEEK NOT COMPLETE WITHIN 120 MS/
9456	044616	052117	041440	046517		
9457	044624	046120	052105	020105		
9458	044632	044527	044124	047111		
9459	044640	030440	030062	046440		
9460	044646	000123				
9461	044650	044122	030461	051057	EM41:	. ASCIZ "RH11/RPO4/5/6 ERROR"
9462	044656	030120	027464	027465		
9463	044664	020066	051105	047522		
9464	044672	000122				
9465	044674	040506	040524	020114	EM46:	. ASCIZ /FATAL WRITE CHECK ERROR/
9466	044702	051127	052111	020105		
9467	044710	044103	041505	020113		
9468	044716	051105	047522	000122		
9469						
9470					. SBTTL	STATUS/ERROR INDICATOR MESSAGES
9471						
9472	044724	043117	046106	047111	MSG814:	. ASCIZ /OFFLINE OR UNSAFE DRIVE REQUESTED/
9473	044732	020105	051117	052440		



9474	044740	051516	043101	020105	
9475	044746	051104	053111	020105	
9476	044754	042522	052521	051505	
9477	044762	042524	000104		
9478	044766	047125	047514	042101	MSGB13: .ASCIZ /UNLOADED DRIVE REQUESTED/
9479	044774	042105	042040	044522	
9480	045002	042526	051040	050505	
9481	045010	042525	052123	042105	
9482	045016	000			
9483	045017	120	051105	044523	MSGB12: .ASCIZ /PERSISTENT UNSAFE/
9484	045024	052123	047105	020124	
9485	045032	047125	040523	042506	
9486	045040	000			
9487	045041	120	051101	052111	MSGB11: .ASCIZ /PARITY ERROR OCCURRED/
9488	045046	020131	051105	047522	
9489	045054	020122	041517	052503	
9490	045062	051122	042105	000	
9491	045067	106	052101	046101	MSGB10: .ASCIZ /FATAL PARITY ERROR/
9492	045074	050040	051101	052111	
9493	045102	020131	051105	047522	
9494	045110	000122			
9495	045112	047523	052106	040527	MSGB09: .ASCIZ /SOFTWARE TIMEOUT ON THIS DRIVE/
9496	045120	042522	052040	046511	
9497	045126	047505	052125	047440	
9498	045134	020116	044124	051511	
9499	045142	042040	044522	042526	
9500	045150	000			
9501	045151	123	043117	053524	MSGB08: .ASCIZ /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
9502	045156	051101	020105	044524	
9503	045164	042515	052517	020124	
9504	045172	047117	040440	047516	
9505	045200	044124	051105	042040	
9506	045206	044522	042526	000	
9507	045213	105	051122	051117	MSGB06: .ASCIZ "ERROR OCCURRED DURING I/O OPERATION"
9508	045220	047440	041503	051125	
9509	045226	042522	020104	052504	
9510	045234	044522	043516	044440	
9511	045242	047457	047440	042520	
9512	045250	040522	044524	047117	
9513	045256	000			
9514	045257	105	051122	051117	MSGB05: .ASCIZ "ERROR OCCURRED DURING NON-I/O OPERATION"
9515	045264	047440	041503	051125	
9516	045272	042522	020104	052504	
9517	045300	044522	043516	047040	
9518	045306	047117	044455	047457	
9519	045314	047440	042520	040522	
9520	045322	044524	047117	000	
9521	045327	125	051516	043101	MSGB04: .ASCIZ /UNSAFE OCCURRED/
9522	045334	020105	041517	052503	
9523	045342	051122	042105	000	
9524	045347	101	052125	046517	MSGB03: .ASCIZ /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
9525	045354	052101	041511	051040	
9526	045362	041505	046101	041111	
9527	045370	040522	042524	051440	
9528	045376	050505	042525	041516	
9529	045404	020105	041517	052503	

9530	045412	051122	042105	000	
9531	045417	104	044522	042526	MSGB02: .ASCIZ /DRIVE HAS NOT RESPONDED TO PORT REQUEST/
9532	045424	044040	051501	047040	
9533	045432	052117	051040	051505	
9534	045440	047520	042116	042105	
9535	045446	052040	020117	047520	
9536	045454	052122	051040	050505	
9537	045462	042525	052123	000	
9538	045467	104	044522	042526	MSGB01: .ASCIZ /DRIVE HAS BECOME NON-EXISTENT/
9539	045474	044040	051501	041040	
9540	045502	041505	046517	020105	
9541	045510	047516	026516	054105	
9542	045516	051511	042524	052116	
9543	045524	000			
9544					
9545					.SBTTL DATA HEADER (DT) MESSAGES
9546					
9547	045525	105	051122	050040	DH1: .ASCIZ /ERR PC RPAS/
9548	045532	020103	051040	040520	
9549	045540	000123			
9550	045542	051105	020122	041520	DH2: .ASCIZ /ERR PC DRIVE RPAS RPDS1 RPER1 RPER2 RPER3/
9551	045550	020040	051104	053111	
9552	045556	020105	020040	050122	
9553	045564	051501	020040	020040	
9554	045572	050122	051504	020061	
9555	045600	020040	050122	051105	
9556	045606	020061	020040	050122	
9557	045614	051105	020062	020040	
9558	045622	050122	051105	000063	
9559	045630	042524	052123	020040	DH3: .ASCIZ /TEST ERR PC ADDRESS DATA/
9560	045636	020040	051105	020122	
9561	045644	041520	020040	042101	
9562	045652	051104	051505	020123	
9563	045660	040504	040524	000	
9564	045665	124	051505	020124	DH4: .ASCIZ /TEST ERR PC ADDRESS GDDATA BDDATA/
9565	045672	020040	042440	051122	
9566	045700	050040	020103	040440	
9567	045706	042104	042522	051523	
9568	045714	043440	042104	052101	
9569	045722	020101	041040	042104	
9570	045730	052101	000101		
9571	045734	050122	051503	020061	DH10: .ASCIZ /RPCS1 ERR PC/
9572	045742	020040	051105	020122	
9573	045750	041520	000		
9574	045753	104	044522	042526	DH11: .ASCIZ /DRIVE ERR PC/
9575	045760	020040	042440	051122	
9576	045766	050040	000103		
9577	045772	042524	052123	020040	DH12: .ASCIZ /TEST ERR PC TST PC DRIVE CYLNR TRACK SECTOR/
9578	046000	020040	051105	020122	
9579	046006	041520	020040	051524	
9580	046014	020124	041520	020040	
9581	046022	051104	053111	020105	
9582	046030	020040	054503	047114	
9583	046036	051104	020040	051124	
9584	046044	041501	020113	020040	
9585	046052	042523	052103	051117	









9698	047166	034354	034356			
9699	047172	001366	001116		DT10:	. WORD RH. ADR, \$ERRPC
9700	047176	001166	001116		DT11:	. WORD \$REG2, \$ERRPC
9701	047202	001176	001116	001162	DT12:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, CYL. DS, TRK. DS, SEC. DS
9702	047210	001254	001270	001274		
9703	047216	001272				
9704	047220	001270	001274	001272	DT12A:	. WORD CYL. DS, TRK. DS, SEC. DS, CYL. RD, TRK. RD, SEC. RD
9705	047226	001262	001264	001266		
9706	047234	001176	001116	001162	DT13:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, CYL. DS, TRK. DS, SEC. DS
9707	047242	001254	001270	001274		
9708	047250	001272				
9709	047252	001124	001126	001172	DT13A:	. WORD \$GDDAT, \$BDDAT, \$REG4, \$GDADR, \$BDADR
9710	047260	001120	001122			
9711	047264	001176	001116	001254	DT17:	. WORD \$TMPO, \$ERRPC, CHKDRV, RP. REG, RP. REG+12, RP. REG+14, RP. REG+40, RP. REG+42
9712	047272	004204	004216	004220		
9713	047300	004244	004246			
9714	047304	001176	001116	001162	DT21:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, CYL. DS, TRK. DS
9715	047312	001254	001270	001274		
9716	047320	001164	001126	001172	DT21A:	. WORD \$REG1, \$BDDAT, \$REG4, \$REG1
9717	047326	001164				
9718	047330	001176	001116	001254	DT23:	. WORD \$TMPO, \$ERRPC, CHKDRV, CYL. DS, RP. REG, RP. REG+10, RP. REG+12
9719	047336	001270	004204	004214		
9720	047344	004216				
9721	047346	004220	004244	004246	DT23A:	. WORD RP. REG+14, RP. REG+40, RP. REG+42, RP. REG+34, RP. REG+36
9722	047354	004240	004242			
9723	047360	001176	001116	001162	DT41:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV
9724	047366	001254				
9725	047370	001176	001116	001162	DT42:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, RP. REG, RP. REG+10, RP. REG+12
9726	047376	001254	004204	004214		
9727	047404	004216				
9728	047406	001176	001116	001162	DT43:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, RP. REG, RP. REG+10, RP. REG+12
9729	047414	001254	004204	004214		
9730	047422	004216				
9731	047424	004220	004244	004246	DT43A:	. WORD RP. REG+14, RP. REG+40, RP. REG+42
9732	047432	001176	001116	001162	DT44:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, CYL. DS, TRK. DS, SEC. DS
9733	047440	001254	001270	001274		
9734	047446	001272				
9735	047450	004204	004214	004216	DT44A:	. WORD RP. REG, RP. REG+10, RP. REG+12, RP. REG+36, RP. REG+34, RP. REG+06
9736	047456	004242	004240	004212		
9737	047464	004220	004244	004246	DT44B:	. WORD RP. REG+14, RP. REG+40, RP. REG+42
9738	047472	001176	001116	001162	DT45:	. WORD \$TMPO, \$ERRPC, \$REGO, CHKDRV, CYL. DS, TRK. DS, SEC. DS
9739	047500	001254	001270	001274		
9740	047506	001272				
9741	047510	004204	004214	004216	DT45A:	. WORD RP. REG, RP. REG+10, RP. REG+12, RP. REG+36, RP. REG+34, RP. REG+06
9742	047516	004242	004240	004212		
9743	047524	004220	004244	004246	DT45B:	. WORD RP. REG+14, RP. REG+40, RP. REG+42, RP. REG+2, RP. REG+4, RP. REG+22
9744	047532	004206	004210	004226		
9745						
9746					. SBTTL	DATA FORMAT (DF) TABLE
9747						
9748	047540	000001			DF 1:	. WORD 1 ; NUMBER OF DATA HEADERS
9749	047542	002			. BYTE 2 ; NUMBER OF WORDS IN DATA TABLE	
9750	047543	000			. BYTE 0 ; ALL 3 NUMBERS ARE OCTAL	
9751						
9752	047544	000001			DF 2:	. WORD 1
9753	047546	007			. BYTE 7	

9754	047547	000		. BYTE	0	
9755						
9756	047550	000001	DF 3:	. WORD	1	
9757	047552	004		. BYTE	4	
9758	047553	000		. BYTE	0	
9759						
9760	047554	000001	DF 4:	. WORD	1	
9761	047556	005		. BYTE	5	
9762	047557	000		. BYTE	0	
9763						
9764	047560	000001	DF 10:	. WORD	1	
9765	047562	002		. BYTE	2	
9766	047563	000		. BYTE	0	
9767						
9768	047564	000001	DF 11:	. WORD	1	
9769	047566	002		. BYTE	2	
9770	047567	000		. BYTE	0	
9771						
9772	047570	000002	DF 12:	. WORD	2	; 2 DH'S TO BE TYPED ; 7 DATA WORDS FOLLOW THE 1ST DH ; WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL ; ADDRESS OF 2ND DH ; 6 DATA WORDS FOLLOW THE 2ND DH ; ALL WORDS ARE OCTAL
9773	047572	007		. BYTE	7	
9774	047573	160		. BYTE	160	
9775	047574	046061		. WORD	DH12A	
9776	047576	006		. BYTE	6	
9777	047577	000		. BYTE	0	
9778						
9779	047600	000002	DF 13:	. WORD	2	; WORD 3 IS DECIMAL
9780	047602	007		. BYTE	7	
9781	047603	160		. BYTE	160	
9782	047604	046140		. WORD	DH13A	
9783	047606	005		. BYTE	5	
9784	047607	004		. BYTE	4	
9785						
9786	047610	000000	DF 14:	. WORD	0	; WORD 3 IS DECIMAL
9787	047612	005		. BYTE	5	
9788	047613	004		. BYTE	4	
9789						
9790	047614	000001	DF 17:	. WORD	1	
9791	047616	010		. BYTE	D8	
9792	047617	000		. BYTE	0	
9793						
9794	047620	000002	DF 21:	. WORD	2	
9795	047622	006		. BYTE	6	
9796	047623	060		. BYTE	60	
9797	047624	046362		. WORD	DH21A	
9798	047626	004		. BYTE	4	
9799	047627	014		. BYTE	14	
9800						
9801	047630	000000	DF 22:	. WORD	0	
9802	047632	004		. BYTE	4	
9803	047633	014		. BYTE	14	
9804						
9805	047634	000002	DF 23:	. WORD	2	; WORD 4 IS DECIMAL
9806	047636	007		. BYTE	7	
9807	047637	010		. BYTE	10	
9808	047640	046507		. WORD	DH23A	
9809	047642	005		. BYTE	5	



9810	047643	000			. BYTE	0
9811						
9812						
9813	047644	000001		DF41:	. WORD	1
9814	047646	004			. BYTE	4
9815	047647	000			. BYTE	0
9816						
9817	047650	000001		DF42:	. WORD	1
9818	047652	007			. BYTE	7
9819	047653	000			. BYTE	0
9820						
9821	047654	000002		DF43:	. WORD	2
9822	047656	007			. BYTE	7
9823	047657	000			. BYTE	0
9824	047660	046700			. WORD	DH43A
9825	047662	003			. BYTE	3
9826	047663	000			. BYTE	0
9827						
9828	047664	000003		DF44:	. WORD	3
9829	047666	007			. BYTE	7
9830	047667	160			. BYTE	160
9831	047670	046726			. WORD	DH44A
9832	047672	006			. BYTE	6
9833	047673	000			. BYTE	0
9834	047674	047003			. WORD	DH44B
9835	047676	003			. BYTE	3
9836	047677	000			. BYTE	0
9837						
9838	047700	000003		DF45:	. WORD	3
9839	047702	007			. BYTE	7
9840	047703	160			. BYTE	160
9841	047704	046726			. WORD	DH44A
9842	047706	006			. BYTE	6
9843	047707	000			. BYTE	0
9844	047710	047031			. WORD	DH45A
9845	047712	006			. BYTE	6
9846	047713	000			. BYTE	0
9847						
9848						
9849		047714			. EVEN BUFFER=.	
9850						
9851	047714	005015	041412	051132	TITLE: . ASCII	<CR><LF><LF>/CZRJAC/<CR><LF>
9852	047722	040512	006503	012		
9853	047727	122	030120	027464	. ASCII2	@RPO4/5/6 MECHANICAL & READ-WRITE TEST@<CR><LF><LF>
9854	047734	027465	020066	042515		
9855	047742	044103	047101	041511		
9856	047750	046101	023040	051040		
9857	047756	040506	026504	051127		
9858	047764	062111	020106	042524		
9859	047772	062123	005015	000012		
9860	050000	005016	047524	052040	LOADRV: . ASCII	<CR><LF>/TO TEST DRIVE 0 REPLACE THE 'XXDP' PACK ON DRIVE 0/<CR><LF>
9861	050006	061506	020124	051104		
9862	050014	063111	020106	020060		
9863	050022	042522	046120	041501		
9864	050030	020106	044124	020106		
9865	050036	054047	042130	023520		

```

9866 050044 050040 041501 020113
9867 050052 047117 042040 044522
9868 050060 042526 030040 005015
9869 050066 044527 044124 040440
9870 050074 047516 044124 051105
9871 050102 050040 041501 026113
9872 050110 041440 042514 051101
9873 050116 046440 046505 051117
9874 050124 020131 047514 040503
9875 050132 044524 047117 032040
9876 050140 026060 040440 042116
9877 050146 051040 051505 040524
9878 050154 052122 005015
9879 050160 044124 020105 051120
9880 050166 043517 040522 006515
9881 050174 000012
9882 050176 005015 054523 052123
9883 050204 046505 044040 051501
9884 050212 030440 045466 046440
9885 050220 046505 051117 026131
9886 050226 023440 054130 050104
9887 050234 020047 047514 042101
9888 050242 051105 053440 046111
9889 050250 020114 042502 047440
9890 050256 042526 053522 044522
9891 050264 052124 047105 005015
9892 050272 000012
9893
9894
9895
9896
9897
9898
9899
9900
9901
9902
9903
9904 050274 010046
9905 050276 010146
9906 050300 013746 000004
9907 050304 013746 000006
9908 050310 010600
9909
9910 050312 104400
9911 050314 012637 000006
9912 050320 012737 050340 000004
9913 050326 012701 020000
9914 050332 005711
9915 050334 005721
9916 050336 000775
9917 050340 162701 000002
9918 050344 010006
9919 050346 012637 000006
9920 050352 012637 000004
9921 050356 010137 050370
    
```

```

        .ASCII /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40, AND RESTART/<<CR><LF>
        .ASCIZ /THE PROGRAM/<<CR><LF>
NOLOAD: .ASCIZ <<CR><LF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER WILL BE OVERWRITTEN/<<CR><L
        .EVEN
        .SBTTL ROUTINE TO SIZE MEMORY
        ;; *****
        ;;CALL:
        ;;      JSR      PC,SSIZE
        ;;      RETURN
        ;;$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
SSIZE:  MOV      RO,-(SP)      ;; SAVE RO ON THE STACK
        MOV      R1,-(SP)      ;; SAVE R1 ON THE STACK
        MOV      @#ERRVEC,-(SP) ;; SAVE PRESENT ERROR VECTOR PS & PC
        MOV      @#ERRVEC+2,-(SP)
        MOV      SP,RO        ;; SAVE THE STACK POINTER
        ;;SET THE ERRVEC PS TO THE PRESENT PS
        TRAP
        MOV      (SP)+,@#ERRVEC+2 ;; SAVE THE PSW IN @#ERRVEC+2
        MOV      #25,@#ERRVEC    ;; SET FOR TIMEOUT
        MOV      #20000,R1       ;; FIRST ADDRESS
1$:     TST      (R1)           ;; TEST THIS ADDRESS
        TST      (R1)+         ;; STEP TO NEXT ADDRESS
        BR      1$             ;; TRY ANOTHER
2$:     SUB      #2,R1          ;; DROP BACK
        MOV      RO,SP         ;; RESTORE THE STACK
        MOV      (SP)+,@#ERRVEC+2 ;; RESTORE ERROR VECTOR
        MOV      (SP)+,@#ERRVEC
        MOV      R1,$LSTAD     ;; LAST ADDRESS
    
```



```

9922 050362 012601          MOV      (SP)+, R1      ;; RESTORE R1
9923 050364 012600          MOV      (SP)+, R0      ;; RESTORE R0
9924 050366 000207          RTS      PC
9925 050370 000000          $LSTAD: .WORD 0        ;; CONTAINS THE LAST ADDRESS
9926
9927                          ;; *****
9928                          .SBTTL GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
9929                          ; THIS ROUTINE IS USED TO ENSURE THE BUS ADDRESS
9930                          ; OF THE RH11/RPO4 IS SETUP TO READ THE PROPER VALUE.
9931                          ; IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
9932                          ; REQUIRED.
9933                          ; NOTE: THIS ROUTINE DESTROYS R0-R4
9934                          ; CALL
9935                          ;
9936                          ; JSR      PC, @GETADR
9937                          ; RETURN
9938
9939 050372 005737 001226      GETADR: TST      @BUSADR  ; INPUT FROM TTY REQUESTED?
9940 050376 001433              BEQ      7$            ; NO--BRANCH
9941 050400 005037 001226      CLR      @BUSADR      ; YES--CLEAR THE REQUEST FLAG
9942 050404 012700 001366      1$: MOV      @RH.ADR, R0 ; FIRST ADDRESS
9943 050410 012703 050560      MOV      @MRPCS1, R3  ; "RPCS1="
9944 050414 011004              MOV      (R0), R4      ; PRESENT RPCS1 ADDRESS
9945 050416 004737 050602      JSR      PC, CLRBF    ; CLEAR INPUT BUFFER
9946 050422 004037 031520      JSR      R0, @GETNUM  ; GET NEW RPCS1
9947 050426 000402              BR      2$            ; COMMA
9948 050430 000765              BR      1$            ; PERIOD
9949 050432 000414              BR      5$            ; DOUBLE PERIOD
9950 050434 010420      2$: MOV      R4, (R0)+   ; SAVE NEW RPCS1
9951 050436 012703 050571      MOV      @MRHVEC, R3  ; "RHVEC="
9952 050442 011004              MOV      (R0), R4      ; PRESENT RH11 VECTOR ADDRESS
9953 050444 004737 050602      JSR      PC, CLRBF    ; CLEAR INPUT BUFFER
9954 050450 004037 031520      JSR      R0, @GETNUM  ; GET NEW RHVEC
9955 050454 000402              BR      3$            ; COMMA
9956 050456 000752              BR      1$            ; PERIOD
9957 050460 000401              BR      5$            ; DOUBLE PERIOD
9958 050462 010420      3$: MOV      R4, (R0)+   ; SAVE NEW RHVEC
9959 050464 010410      5$: MOV      R4, (R0)   ; SAVE INPUT
9960 050466 013701 000004      7$: MOV      @ERRVEC, R1 ; SAVE THE ERROR VECTOR
9961 050472 012737 050526 000004 MOV      @R$, @ERRVEC ; SETUP FOR TRAP
9962 050500 005777 130662      TST      @RH.ADR     ; CHECK FOR RH11/RPO4
9963 050504 010137 000004      MOV      R1, @ERRVEC ; RESTORE ERROR VECTOR
9964 050510 012700 001366      MOV      @RH.ADR, R0 ; FIRST ADDRESS OF NEW PARAMETERS
9965 050514 012701 034516      MOV      @RPADR, R1  ; FIRST ADDRESS OF WHERE TO PUT THEM
9966 050520 012021              MOV      (R0)+, (R1)+ ; BUS ADDRESS
9967 050522 012021              MOV      (R0)+, (R1)+ ; VECTOR ADDRESS
9968 050524 000207          RTS      PC          ; RETURN
9969 050526 010137 000004      8$: MOV      R1, @ERRVEC ; RESTORE ERROR VECTOR
9970 050532 022626          CMP      (SP)+, (SP)+ ; CLEAN OFF THE STACK
9971 050534 104010          ERROR 10          ; REPORT THE ERROR
9972 050536 005737 000042      TST      @#42        ; IS THERE A MONITOR?
9973 050542 001720          BEQ      1$          ; NO--GO ASK FOR ADDRESS
9974 050544 005037 001232      CLR      @DRVSEL     ; YES--NO DRIVES SELECTED
9975 050550 005037 017542      CLR      @SEOPCT    ; NO PASSES
9976 050554 000137 017366      JMP      @SEOP      ; GO TO END OF PROGRAM
9977

```

```

9978 050560 005015 050122 051503 MRPCS1: .ASCIZ <CR><LF>/RPCS1=/
9979 050566 036461 000
9980 050571 015 051012 053110 MRHVEC: .ASCIZ <CR><LF>/RHVEC=/
9981 050576 041505 000075
9982 ;*****
9983 ;ROUTINE TO CLEAR INPUT BUFFER FOR NEW CS1 AND VEC
9984 ;
9985 ;
9986 050602 010046 CLRBF: MOV RO,-(SP) ;SAVE RO
9987 050604 010146 MOV R1,-(SP)
9988 050606 012701 000010 MOV #10,R1 ;COUNT 10 LOCATIONS
9989 050612 012700 022522 MOV #$TTYIN,RO ;START ADR OF INPUT BUF
9990 050616 005020 1$: CLR (RO)+
9991 050620 005301 DEC R1
9992 050622 001375 BNE 1$
9993 050624 012601 MOV (SP)+,R1
9994 050626 012600 MOV (SP)+,RO ;RESTORE RO
9995 050630 000207 RTS PC
9996 000001 .END
  
```



ABS = 000200	1807#													
ACL = 000040	1841#	1850#												
ACTDRV 034434	7959#	8238#	8289#	8602#	8611#	8839								
ACTSTR 034435	7965#	8841#	8856#											
ACU = 100000	1796#													
AOE = 001000	1713#													
ATA = 100000	1700#													
ATABIT 034504	3286	7305	7311	8031#	8204	8301	8406	8765	8784	8792	8809	8915		
ATO = 000001	1733#													
AT1 = 000002	1734#													
AT2 = 000004	1735#													
AT3 = 000010	1736#													
AT4 = 000020	1737#													
AT5 = 000040	1738#													
AT6 = 000100	1739#													
AT7 = 000200	1740#													
A16 = 000400	1631#													
A17 = 001000	1632#													
BA1 = 000010	1649#													
BITS 001424	2014#	3447	3481	3519	3562	3602	3682	3725	3775	3875	4032	4098	4202	
	4292	4390	4504	4575	4704	4750	4824	4913	7372	7403	7420	7473	7647	
BIT0 = 000001	1606#													
BIT00 = 000001	1596#	1606	2014	2030	3257	3274	3296	3349	6119					
BIT01 = 000002	1595#	1605	2015	2031	3260	6445	6473	8281	8528					
BIT02 = 000004	1594#	1604	2016	2032	6182	6445	6476							
BIT03 = 000010	1593#	1603	2017	2033	6482	8724	9037							
BIT04 = 000020	1592#	1602	2018	2034	6479	8759								
BIT05 = 000040	1591#	1601	2019	2035	3918	3934	4010	4140	4154	4169	4174	4239	4263	
	4268	4327	4345	4362	4367	4425	4443	4460	4465	6482	6592	8121	8548	
	8739	8889												
BIT06 = 000100	1590#	1600	2020	2036	6485	8213	8272	8346	8637	9098				
BIT07 = 000200	1589#	1599	2021	2037	8213	8480	8623	8724	8739	8759	8788	9070		
BIT08 = 000400	1588#	1598	2022	6473	8213	8902								
BIT09 = 001000	1587#	1597	2023	5040	5665	6488	8887							
BIT1 = 000002	1605#													
BIT10 = 002000	1586#	2024	5024	6131	6445	6476	7666	8530						
BIT11 = 004000	1585#	2025	5672	6476	8168	8305	8323	8499	8810					
BIT12 = 010000	1584#	2026	6445	6479	8163	8195	8213	8283	8317	8495	8514	8526	8539	
	8749	8751	8975	9032	9099									
BIT13 = 020000	1583#	2027	5031	6445	6473	8266	8697	8978						
BIT14 = 040000	1582#	2028	3929	4135	4149	4234	4322	4340	4420	4438	5647	6373	6445	
	6479	8278	8314	8684	8826	8942	8987	8991	9096					
BIT15 = 100000	1581#	2029	6637	6638	8266	8278	8281	8283	8314	8317	8499	8528	8530	
	8637	8724	8739	8749	8759	8826	8887	8902	8942	8949				
BIT2 = 000004	1604#	8949												
BIT3 = 000010	1603#													
BIT4 = 000020	1602#													
BIT5 = 000040	1601#													
BIT6 = 000100	1600#	8307												
BIT7 = 000200	1599#													
BIT8 = 000400	1598#													
BIT9 = 001000	1597#													
BPTVEC = 000014	1613#													
BUFFER = 047714	2671	2695	2719	2743	3895	4522	4537	4550	4593	4603	4617	4738	4840	
	4864	4872	4880	4886	4894	6556#	6557	6559	6561	6562	6563	6636	6673	
	6816	6835	6836	6860	6949	6981	7024	7066	7083	7084	7101	9849#		







DFLT	001664	2106#	6108											
DF1	047540	2832	9748#											
DF10	047560	2896	9764#											
DF11	047564	2906	9768#											
DF12	047570	2919	9772#											
DF13	047600	2932	2953	9779#										
DF14	047610	2940	2961	9786#										
DF17	047614	2971	2981	9790#										
DF2	047544	2842	2872	9752#										
DF21	047620	2994	9794#											
DF22	047630	3002	9801#											
DF23	047634	3014	3026	9805#										
DF3	047550	2852	9756#											
DF4	047554	2862	9760#											
DF41	047644	3059	9813#											
DF42	047650	3069	9817#											
DF43	047654	3081	9821#											
DF44	047664	3096	9828#											
DF45	047700	3111	3126	9838#										
DH1	045525	2830	9547#											
DH10	045734	2894	9571#											
DH11	045753	2904	9574#											
DH12	045772	2917	2930	2951	3094	3109	3124	9577#						
DH12A	046061	9587#	9775											
DH13A	046140	9595#	9782											
DH17	046206	2969	2979	9602#										
DH2	045542	2840	2870	9550#										
DH21	046304	2992	9613#											
DH21A	046362	9621#	9797											
DH23	046421	3012	3024	9627#										
DH23A	046507	9637#	9808											
DH3	045630	2850	9559#											
DH4	045665	2860	9564#											
DH41	046554	3057	9644#											
DH42	046612	3067	3079	9649#										
DH43A	046700	9658#	9824											
DH44A	046726	9662#	9831	9841										
DH44B	047003	9670#	9834											
DH45A	047031	9674#	9844											
DIGB =	000004	1687#												
DISPLA	001142	1919#	3168*	3176*	3454*	3488*	3526*	3569*	3609*	3689*	3732*	3782*	3882*	4039*
		4105*	4209*	4299*	4397*	4511*	4582*	4711*	4831*	4920*	5023*	5686*		
DISPRE	000174	1486#	3176											
DLT =	100000	1661#												
DL64 =	000020	1689#												
DMD =	000001	1723#												
DORTI	027054	3916	4129	4227	4315	4413	6623#							
DPB. A	004104	2666#	3342*	3348*	3351*	3352*	3354*	3457*	3786*	3804*	3811	3887*	3890*	3899
		3920	3921	3931	3992*	3999*	4001	4003	4007	4925*	4926*	4930*	4931*	6227
		6229	6233	6234	6235	6236	6244	6570*						
DPB. B	004124	2690#	3343*	3458*	3459*	3460*	3491*	3493*	3495*	3530*	3531*	3533*	3538*	3539
		3541*	3546*	3547	3573*	3574*	3576*	3583*	3613*	3614*	3624*	3630*	3646*	3652*
		3693*	3694*	3700*	3702*	3736*	3737*	3746*	3748*	3750*	3752*	3754*	3756*	3785*
		3787*	3792*	3802*	3804	3831*	3832	3834*	4043*	4045*	4047*	4058*	4059	6211*
		6214*	6258	6260	6264	6265	6266	6267	6279	6282	6285			
DPB. C	004144	2714#	3344*	3492*	3494*	3496*	4044*	4046*	4048*	4054*	4055	4057*	6212*	6215*









EXIT20	016436	4739	4748	4794#										
EXIT21	017214	4835	4898#											
EXIT22	017364	4935#												
EXIT3	007304	3575	3588#											
EXIT4	007672	3615	3665#											
EXIT5	010072	3695	3707	3710#										
EXIT6	010336	3738	3745	3760#										
EXIT7	010716	3789	3833	3835#										
EXT1 =	000001	1763#												
EXT10 =	000010	1766#												
EXT2 =	000002	1764#												
EXT20 =	000020	1767#												
EXT4 =	000004	1765#												
EXT40 =	000040	1768#												
FC	001510	2043#	3495	3533	3547	3576	3620	3630	3663	3696	3708	3739	3792	3793
		3800	3890	3891	3892	3989	4047	4048	4057	4115	4222	4335	4433	4518
		4590	4742	4838	4930	7165	7183							
FEN =	000200	1789#												
FER =	000020	1708#												
FILBUF	027706	4517	4588	6814#										
FILRAM	030770	4844	7066#											
FMT22 =	010000	1825#	1889#	3348	6556									
FS	001524	2049#	3458	3491	3530	3573	3613	3693	3736	4043	4044	4116	4591	4718
		4732	4737											
FT	001516	2046#	3459	3493	3531	3574	3614	3694	3737	3785	3834	3889	4045	4046
		4117	4519	4752										
F1 =	000002	1676#												
F2 =	000004	1677#												
F3 =	000010	1678#												
F4 =	000020	1679#												
F5 =	000040	1680#												
GETADR	050372	3215	9939#											
GETNUM	031520	7209	7231#	9946	9954									
GETREG=	000141	1882#												
GETREQ	042672	8302	8523	8678	8714	8775	8825	8886	8901	8939	8946	9227#		
GETSMR	031426	3220	7201#											
GNS =	##### U	1485	4950	4956	4964	4981	5096	5113	5768	5769	5770	5771	5772	5774
		5776	5777	5778	5779	5780	7205	7285	7318	7477	7583			
GO =	000001	1675#												
GRV =	000010	1688#												
GTSMR =	104406	3209	5774#											
GTTST1	032014	7310	7315#	7364	7377	7388	7391	7417	7424	7433	7439	7448	7465	
GTTST2	032100	7332#	7428											
GTTST3	032442	7335	7339	7343	7347	7351	7410#							
GTTST4	032444	7353	7374	7408	7411#									
GTTST5	032554	7412	7434#											
GTTST6	032614	7436	7445#											
GT. PR1	031656	3278	7281#											
GT. PR2	031660	7282#	7290	7291	7292	7293	7294	7300	7301					
	032010	7312#	7327	7444	7453	7646								
HCE =	000200	1711#												
HCI =	002000	1823#												
HCRC =	000400	1712#												
HT =	000011	1519#	5188	5229										
IAE =	002000	1714#												
IC	001514	2045#	3538	3546	3580	3640	3662	3704	3705	3758	3893	3980	3985	3987













































	5756#	5767	5770#	5771#	5772#	5773#	5774	5775#	5776	5777#	5778#	5779#	5780#
STRAP2 023160	5756#	5767											
STRP = 000014	5760#	5769#	5770#	5771#	5772#	5773#	5774	5775#	5776	5777#	5778#	5779#	5780#
	5781#												
STRPAD 023172	5750	5767#											
STSTNM 001102	1900#	3450#	3454	3484#	3488	3522#	3526	3565#	3569	3605#	3609	3685#	3689
	3728#	3732	3778#	3782	3878#	3882	4035#	4039	4101#	4105	4205#	4209	4295#
	4299	4393#	4397	4507#	4511	4578#	4582	4707#	4711	4827#	4831	4916#	4920
	4969#	5023	5053	5064	5637	5681#	5686	5690	6165	6367			
	5573	5574	5586	5604	5618	5622#	9989						
STTYIN 022522	5773												
STYPBN= ##### U	5320#	5772											
STYPDS 021112	5176#	5760	5768										
STYPE 020444	5197	5204	5211	5216#	5217	5521							
STYPEC 020614	5222	5224	5227#										
STYPEX 020662	5260#	5769											
STYPOC 020710	5259	5262#	5771										
STYPON 020724	5255#	5770											
STYPOS 020664	5650#												
SXTSTR 022622	4986#												
SSGET4= 000000	5256#	5260#	5270	5305#									
SOF ILL 021107	5033	5647											
S4OCAT= ##### U	1481#	1485#	1493	1494#	1496#	1498#	1501#	1897#	1945	3147	3161	3162	3534
= 050632	3542	3577	3581	3622	3644	3698	3742	3806	3809	3904	3942	4523	4528
	4531	4539	4546	4595	4605	4622	4628	4714	4768	4774	4784	4834	4846
	4867	4873	4881	4887	4895	4951#	4982#	4994	4995#	5053	5097#	5229	5374#
	5379	5383#	5384	5385	5622#	5623	5630#	5689	5690	5862#	7478#	9151#	9152#
	9153#	9154#	9155#	9156#	9157#	9158#	9159	9849					



CKCHR	3128#	7239	7289	7516	7553	7593	7756	7769	7780	7831				
CKDIG	3128#	7297	7535											
CKNUM	3128#	7247	7568	7601										
COMEN	1#	1621#	3385	3420	4066	4475								
COMND	3128#	3354	3457	3786	3886	4926	4931							
DO	3128#	3353	3355	3464	3465	3500	3501	3537	3545	3579	3584	3625	3631	3647
	3701	3703	3747	3749	3751	3753	3755	3757	3808	3829	4051	4053	4927	4932
DODTA	3128#	4526	4530	4535	4545	4552	4598	4609	4625	4631	4771	4777	4787	4848
	4875	4883	4889	4897										4869
DRV. IN	3128#	4173	4267	4366	4464									
ENDCOM	1#	1621#	3413	3431	4079	4484								
ENDPAS	3128#	4979												
ERRCAL	7851#	8653	8704	8707	8982	9043								
ERREND	3128#	5047												
ERROR	1515#	3339	3933	4009	4143	4157	4172	4242	4266	4330	4348	4365	4428	4446
	6239	6240	6241	6242	6243	6270	6271	6272	6273	6278	6313	6314	6315	6316
	6362	6363	6364	6365	6366	6392	6393	6394	6395	6396	6418	6419	6420	6421
	6569	6610	6611	6612	6613	6614	6926	6931	7042	7048	7106	7107	8654	8705
	8984	9044	9971											8709
ERRTYP	3128#	5015												
ER. NDX	3128#	6233	6264	6307	6356	6386	6412	6604						
ESCAPE	1#	1621#	4119	4220	4310	4408	6232	6263	6306	6354	6385	6411	6567	6603
	6929	7041	7046	7094										6925
GETPRI	1#	1621#	5950	9910										
GETSHR	1#	1426#	1621#	3204										
LOOP	3128#	3534	3542	3577	3581	3622	3644	3698	3742	3805	3809	3903	3941	4523
	4531	4539	4546	4595	4605	4622	4628	4768	4774	4784	4846	4867	4873	4881
	4895													4887
MORETA	1891#	1946												
MORE. S	3128#	3445	3479	3517	3560	3600	3680	3723	3773	3873	4030	4096	4200	4290
	4502	4573	4702	4822	4911									4388
MSG	3436#	3438	3468#	3470	3504#	3506	3551#	3553	3590#	3592	3667#	3669	3712#	3714
	3764	3837#	3840	4017#	4019	4083#	4086	4189#	4191	4277#	4279	4375#	4377	4489#
	4559#	4561	4640#	4642	4797#	4799	4902#	4904						4491
MULT	1#	1621#												
NEWTST	1#	1621#	3436	3468	3504	3551	3590	3667	3712	3762	3838	4017	4084	4189
	4375	4489	4559	4640	4797	4902								4277
POP	1#	1621#	5361	5729	5918	6001	6146	6195						
PUSH	1#	1621#	5320	5709	5899	5953	6101	6159						
REPORT	1#	1621#	3128#	4180	4183	4272	4371	4469						
RPO4. D	2#	7851												
SAV. RH	3128#	4137	4151	4166	4236	4260	4324	4342	4359	4422	4440	4457		
SCOPE	1516#	3466	3502	3549	3588	3665	3710	3760	3835	4015	4061	4186	4274	4373
	4557	4637	4794	4898	4935									4471
SETPRI	1#	1621#	5551											
SETTRA	5760#	5769	5770	5771	5772	5774	5776	5777	5778	5779	5780			
SETUP	1#	1621#	3142											
SET. TN	3128#	3446	3480	3518	3561	3601	3681	3724	3774	3874	4031	4097	4201	4291
	4503	4574	4703	4823	4912									4389
SKIP	1#	1621#												
SLASH	1#	1621#												
SPACE	1621#													
STARS	1#	1491	1621#	1625	1667	1671	1862	1893	1945	3030	3436	3444	3468	3478
	3516	3551	3559	3590	3599	3667	3679	3712	3722	3762	3772	3838	3872	4017
	4084	4095	4189	4199	4277	4289	4375	4387	4489	4501	4559	4572	4640	4701
	4821	4902	4910	4941	5001	5054	5161	5232	5310	5378	5454	5469	5540	5564





.S40CA 18  
.1170 18

.ABS. 050632 000

ERRORS DETECTED: 0

DSKZ: CZRJAC. BIN, DSKZ: CZRJAC. LST/CRF/SOL=DSKZ: CZRJAC. SML, DSKZ: CZRJAC. 010, DSKZ: CZRJAC. P11  
RUN-TIME: 27 37 3 SECONDS  
RUN-TIME RATIO: 315/68=4.5  
CORE USED: 52K (103 PAGES)