RP11E

DRIVE POSITIONING TEST

MD-11-DZRPZ-B

EP-DZRPZ-B-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A

```
                                .REM    %
```

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

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

PRODUCT NAME: RP11E DRIVE POSITIONING TEST

DATE CREATED: MAY 21, 1976

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: C. HESS

CONTENTS

1. ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT WILL ENSURE THAT THE DISK DRIVE IS CAPABLE OF PERFORMING SEEKS, THAT THE ACCESS TIMES ARE WITHIN TOLERANCE, AND THAT THE CYLINDER ADDRESSING CAPABILITY OF THE DRIVE IS FUNCTIONING PROPERLY.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH 8K OF MEMORY (WITH OR WITHOUT HARDWARE SWITCH REGISTER); CONSOLE TELETYPE; RP11E DISK CONTROLLER;
1 TO 8 RP02, RPR02, OR RP03 DISK DRIVES WITH FORMATTED PACKS.

2.2 STORAGE

THIS PROGRAM WILL LOAD AND RUN IN 8K.

2.3 PRELIMINARY PROGRAMS

MAINDEC-11-DZRPW - RP11E DISKLESS LOGIC TEST

3. LOADING PROCEDURE

*** BEFOR STARTING REFER TO SECTION 5.***
THE PROGRAM MAY BE LOADED FROM EITHER PAPER TAPE OR AN 'XXDP' DEVICE. REFER TO EITHER THE STANDARD PROCEDURES FOR LOADING 'ABS' FORMAT PAPER TAPES OR TO THE 'XXDP' SYSTEM REFERENCE DOCUMENT.

THIS PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN. IF THE PROGRAM IS PART OF A CHAIN AND IS LOADED FROM AN RP02, RPR02, OR AN RP03, THE PROGRAM WILL BYPASS TESTING DRIVE 0.

4. STARTING PROCEDURE

4.1 STARTING ADDRESSES

200	NORMAL STARTING ADDRESS
204	SELECT OPERATING PARAMETERS
210	SELECT RP11 ADDRESS
214	COMBINATION OF 204 AND 210

4.1.1 START FROM LOCATION 200

```
                                WHEN THE PROGRAM IS STARTED FROM LOCATION 200, THE PROGRAM WILL
                                TEST ALL AVAILABLE DRIVES (SEE SECTION 2.1 FOR DRIVE 0 EXCEPTION)
                                USING TESTS 0 - 10, & 12, WITH DEFAULT PARAMETERS.  PREVIOUS
                                PARAMETER CHANGES ARE OVERLAYED.
```

4.1.2 START FROM LOCATION 204

WHEN THE PROGRAM IS STARTED FROM LOCATION 204, THE PARAMETERS ARE RESET TO THEIR DEFAULT VALUES, AND THE PROGRAM ENTERS CONVERSATIONAL MODE. REFER TO SECTION 4.3.2.

4.1.3 START FROM LOCATION 210

WHEN THE PROGRAM IS STARTED FROM LOCATION 210, OPERATION IS THE SAME AS THE START FROM 200, EXCEPT THAT THE PROGRAM ASKS FOR A NEW RP11E ADDRESS, VECTOR ADDRESS, AND PRIORITY. THE OPERATOR MAY ENTER NEW ADDRESS VALUES OR RETAIN THE OLD VALUES. THE TEST PARAMETERS ARE RESET TO THE DEFAULT VALUES. REFER TO SECTION 4.3.1.

4.1.4 START FROM LOCATION 214

PROGRAM START FROM LOCATION 214 IS THE SAME AS THE START FROM LOCATION 204 EXCEPT THAT THE OPERATOR MAY ENTER NEW RP11E ADDRESS, VECTOR ADDRESS, OR PRIORITY. THE TEST PARAMETERS ARE RESET TO THE DEFAULT VALUES. REFER TO SECTIONS 4.3.1 & 4.3.2.

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.
4. SET 'FORMAT' AND 'MAINTENANCE' SWITCHES ON THE RP11E CONTROL PANEL TO 'NORMAL'.
5. LOAD ADDRESS 200, 204, 210, OR 214.
6. SET SWITCHES (SEE SECTION 5.)
7. PRESS START.

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

AN INPUT TERMINATOR: WHEN A PERIOD IS TYPED AFTER A "STATEMENT TERMINATOR PERIOD" IT TELLS

```
  212                                                   THE PSI THIS IS THE END OF ALL CHANGES (LEGAL
  213                                                   ON ALL LINES) BEGIN EXECUTION.
  214
```

```
                        <,><CR>         COMMA

                                        A CONTINUATION INDICATOR AND FIELD SEPARATOR:

                                        1)  THIS IS THE CONSTRUCTION USED WHEN NOT USING
                                            A PERIOD; THAT IS TO SAY THE COMMA SAYS
                                            "TERMINATE THE LINE BUT NOT THE INPUT FOR
                                            PARAMETER CHANGES".

                                        2)  SEPARATE MULTIPLE ENTRIES ON THE SAME TEXT
                                            STRING, E.G., A,B,C,D.

                        </>             SLASH

                                        A MODIFICATION INDICATOR: AT ANY GIVEN PARAMETER
                                        STOP IF A SLASH IS TYPED IT SAYS "OPEN THIS
                                        PARAMETER FOR CHANGES".

                        <^U>            CONTROL-U

                                        DUMP 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   ADDRESS SELECTION
            ----------------

            STARTING THE PROGRAM FROM 200 (8) WILL RESULT IN AUTOMATIC
            SELECTION OF THE DEFAULT RP11E VALUES OF BUS ADDRESS, VECTOR
            ADDRESS, AND PRIORITY LEVEL.  IF THE DEFAULT VALUE OF THE BUS
            ADDRESS DOES NOT RESPOND (TIMES OUT) TO ADDRESSING, IT IS
            REPORTED AS AN ERROR.  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 EITHER LOCATION 210 OR LOCATION 214
            ALLOWS THE OPERATOR TO CHANGE THE RP11E BUS ADDRESS, THE VECTOR
            ADDRESS, AND (OR) THE PRIORITY.
```

```
                                4.3.1.1 ADDRESS SELECTION EXAMPLES

                                        EXAMPLE #1

                                          RPOS=176710/176300..

                                        EXAMPLE #2

                                          RPOS=176710/176300,
                                          RHVEC=254/260,
                                          RHPRIO=5/6..

                                        EXAMPLE #3

                                          RPOS=177200,
                                          RHVEC=254/260..

                                        EXAMPLE #4

                                          RP11 FAILED TO RESPOND TO ADDRESSING
                                          RPOS   ERR PC
                                          176710  XXXXXX
                                          RPOS=176710/176300..

                                        EXAMPLE #5

                                          RPOS=176710/1776\67\6300,
                                          RHVEC=254,
                                          RHPRIO=5,
                                          RPOS=176300

                        4.3.2   DRIVE, TEST, AND PARAMETER SELECTION
                                ----------------------------------

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

                                STARTING THE PROGRAM AT 213 OR 214 ALLOWS THE OPERATOR
                                TO SELECT THE DRIVE(S) TO BE TESTED, THE TESTS TO BE EXECUTED,
                                AND THE PARAMETERS TO USE.
```

```
                        4.3.2.1 DRIVE, TEST, AND PARAMETER SELECTION DESCRIPTION

                        THE FOLLOWING IS A TABLE OF TERMS USED IN THE DESCRIPTION.

                                "R"     REPEATS (ITERATIONS)
                                "FC"    FIRST CYLINDER ADDRESS
                                "LC"    LAST CYLINDER ADDRESS
                                "IC"    INCREMENT CYLINDER
                                "TK"    TRACK ADDRESS

                        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 OPENED, THEY WILL RECIEVE THE PARAMETERS OF
                                THE TEST THAT IS OPEN WHEN <..> IS TYPED.

                                <,> IS ILLEGAL AS A "TEST" LINE TERMINATOR
                                UNLESS A TEST IS TO BE OPENED.

                          ON STARTING THE PROGRAM ALL TESTS WILL RUN AGAINST ALL DRIVES
                        IN A WORST CASE MANNER.  BUT IF THE USER DESIRES TO SELECT THE
                        DRIVES TO BE TESTED, THE TESTS TO BE EXECUTED, AND THE PARAMETERS
                        TO BE USED HE ENTERS CONVERSATION MODE BY TYPING CONTROL C (↑C).

                        THE PROGRAM WILL RESPOND WITH:

                                DRIVE(S)=

                        WE WILL ASSUME FOR OUR EXAMPLE THAT THE OPERATOR WISHES 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<,>" WHICH SAYS "THIS IS THE END OF ANY
                        CHANGES TO THIS LINE BUT TAKE ME DEEPER FOR MORE CHANGES", AND <CR> "TAKE IT."

                        FOR CLARITY THE LINE WILL BE REPEATED AS IT LOOKS AFTER USER RESPONSE WITH
                        THE LINE BELOW IT ADDED, THE PROGRAMS RESPONSE:

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

                        NOW THE USER INSERTS THE TEST NUMBERS HE DESIRES. IN THIS CASE
                        HE WANTS TESTS 2 THRU 7 AND TEST 11 SO HE TYPES 2-7<,> (TO
                        SEPERATE FIELD), 11<..> (END OF ANY CHANGES START EXECUTION),
                        <CR> (TAKE IT).

                        IT NOW LOOKS LIKE THIS

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

```
                                IN OUR NEXT EXAMPLE WE WILL ASSUME THE USER WISHES TO TEST DRIVE
                                4 AND RUN TESTS 1 AND 3 THRU 11 WITH THE PARAMETER FOR TESTS 3
                                AND 10 MODIFIED.
                                THIS IS HOW HE WOULD RESPOND:

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

                                THE USER NOW WILL INSERT A LINE TO GET TESTS 1 AND 3 THRU 11
                                BUT REMEMBER HE WISHES TO "OPEN" TESTS 3 AND 10 FOR PARAMETER
                                CHANGES.  THE ENTIRE ENTRY IS SHOWN BELOW:

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

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

                                THE PSI SCANS THE TEST STRING AND DETERMINES THAT TEST #1'S
                                PARAMETERS WILL REMAIN THE SAME BUT
                                TEST 3'S MUST BE CHANGED.  THEREFORE, TEST
                                3 IS OPENED FOR CHANGE.

                                RESPONDS: <FOR CLARITY ENTIRE TRANSACTION REPEATED>

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

                                AS IN THE NORMAL MANNER TYPING A </> WILL OPEN THIS LINE FOR
                                MODIFICATION OF THE ITERATION COUNT, ENDING THE LINE WITH A
                                <.> (PERIOD) WOULD TERMINATE THE CHANGES FOR THIS TEST, AND AS
                                IN THE EXAMPLE SHOWN USING A <,> (COMMA) WILL GET US 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
```

THE USER DOES NOT WISH TO CHANGE "FC" SO THE FOLLOWING ACTION IS TAKEN:

```
        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=202
```

THE PROGRAM RESPONDS WITH THE PREVIOUSLY ASSIGNED PARAMETER FOR LAST CYLINDER ADDRESS IN THIS CASE USING '202' AS THE EXAMPLE. THIS IS WHAT THE USER INTENDED TO MODIFY AND IS WHY HE OPENED TEST #3. HE WANTS TO CHANGE IT TO CYLINDER 20. THEREFORE HE TYPES </> (OPEN LINE FOR CHANGE), "20" (THE NEW VALUE), <.> (THIS IS THE END OF CHANGES FOR THIS DRIVE), <CR> (TAKE IT).
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= 202/20.<CR>
          TEST 10
            R=1
```

THE PROGRAM HAS LOADED TEST #3 WITH ITS NEW PARAMETERS, TESTS NUMBER 4 THRU 7 WITH THEIR PREVIOUSLY ASSIGNED PARAMETERS AND IS NOW WAITING FOR CHANGES TO TEST 10.

THE USER TYPES </> (OPEN THIS LINE FOR CHANGE), "10" (MAKE ITERATION COUNT 10), <.> (THIS IS THE END OF CHANGES TO THIS TEST) <CR> (TAKE IT).

THE PROGRAM NOW LOADS TEST 10 WITH THE NEW PARAMETERS, TEST 11 WITH PREVIOUSLY ASSIGNED PARAMETERS AND RESPONDS WITH:

```
        DRIVE(S)=
```

SINCE THE USER DID NOT END THE CONVERSATION MODE WITH A <..> 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 CHANGES MADE WE MUST NOTIFY THE PROGRAM WE ARE FINISHED BY TYPING <..>.

```
                                4.3.3.2 DRIVE, TEST, 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 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=202/50..<CR>     ;TEST 10 IS STILL PENDING
                                                                ;AND WILL THEREFORE BE
                                                                ;SET TO THE SAME PARAMETERS
                                                                ;AS TEST 7 -- START EXECUTION


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

                                        DRIVE=0,1,4,            ;TEST DRIVES 0,1, AND 4 IN SEQUENCE
                                        TEST=0-5/               ;CHANGE TEST 0-5
                                          TEST 0
                                            R=10,
                                            FC=0,
                                            LC=202/1..          ;CHANGE LAST CYLINDER FROM 202
                                                                ;TO 1 FOR TEST #0-5; START TESTING

                        4.4     'CONTROL C' OPERATION
                                --------------------

                                THE OPERATOR MAY TERMINATE THE PROGRAM AT ANY TIME BY TYPING A
                                'CONTROL C'.  'CONTROL C' RETURNS THE PROGRAM TO THE PARAMETER
                                ENTRY ROUTINE.  THE TEST PARAMETERS ARE NOT RESTORED TO THE DEFAULT
                                VALUES BY THE 'CONTROL C' RETURN.

                                CONTROL G OPERATION (REFER TO SECTION 5.)
```

```
503
504
```

5. OPERATING PROCEDURE

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G (↑G); THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U (↑U) IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

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(06)=1...TYPE ALL RP11 REGISTERS IF ERROR
SW(04)=1...TYPE THE LONG SEEK TIME GRAPH (TEST #12)
SW(03)=1...PERFORM STALL BETWEEN SEEK ORDERS
SW(02)=1...USE RANDOM STALL VALUE BETWEEN SEEK ORDERS
SW(01)=1...PERFORM INCREMENTING STALL IN TEST 4
SW(00)=1...DO NOT CHECK POSITION AFTER SEEK ORDERS

```
561
562
563
```

6. ERRORS

THERE ARE A NUMBER 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 TYPE OUT WILL CONTAIN THE FOLLOWING:

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

6.1 ERROR TYPES

THE ERRORS THAT OCCUR IN THIS PROGRAM FALL INTO TWO
(2) CATEGORIES DEFINED AND EXPLAINED AS FOLLOWS:

6.1.1 NON-FATAL ERROR

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.

6.1.2 FATAL ERROR

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 NON-FATAL ERROR

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

6.2.2 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

7.1 SUBSYSTEMS WITH MIXED DRIVE TYPES

THE PROGRAM ASSUMES THAT ONLY ONE KIND OF DRIVE IS ATTACHED TO
A SYSTEM (E.G. RP02'S & RPR02'S OR RP03'S). IF BOTH RP02'S

```
                        (AND RPRD2'S) OR RP03'S ARE ON THE SAME SYSTEM, THE PROGRAM WILL
                        ASSUME THAT ALL OF THE DRIVES ARE RP03'S.  TO TEST DRIVES ON A
                        MIXED SYSTEM, TEST ALL OF THE SAME KIND OF DRIVE AT ONE TIME.  THE
                        OTHER DRIVES ON THE CONTROLLER MUST BE POWERED DOWN.
```

7.2 FORMATTED PACKS

EACH OF THE DRIVES BEING TESTED MUST HAVE A FORMATTED PACK. (UNLESS THE PROGRAM IS RUN WITH SWITCH (DD) SET.)

8. MISCELLANEOUS

8.1 EXECUTION TIME

TO MAKE ONE PASS OF THE PROGRAM (TESTS 0 - 7) TAKES APPROXIMATELY 1.8 HOURS FOR EACH RP02 OR RPR02 TESTED AND APPROXIMATELY 6.5 HOURS FOR EACH RP03 TESTED.

NOTE THAT TEST 7 REQUIRES 1.3 HOURS FOR RP02 DRIVES AND 5.4 HOURS FOR RP03 DRIVES. IN SPITE OF THE LENGTHY RUN TIME OF TEST 7, IT IS RECOMMENDED THAT TEST 7 BE RUN OCCASIONLLY TO VERIFY THE SUBTRACTER IN THE DISK DRIVE. TEST 7 HAS ALSO PROVEN TO BE A VALUABLE LONG TERM SEEK VERIFIER, ALSO.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100 AND EXTENDS DOWNWARD IN MEMORY.

8.3 TIMING TEST PRINTOUTS

8.3.1 TIMING TEST (TESTS 10 & 11) PRINTOUT EXAMPLES

EXAMPLE #1

ONE CYLINDER SEEK TIMES
(SEEK TIME IS IN MICRO SECONDS)

FORWARD # OF SEEKS	FORWARD SEEK TIME	REVERSE # OF SEEKS	REVERSE SEEK TIME
193	10000	60	10000
9	12500	142	12500

EXAMPLE #2

SEEK TIMES BETWEEN CYLINDERS 0 & 202
(SEEK TIME IS IN MICRO SECONDS)

FORWARD # OF SEEKS	FORWARD SEEK TIME	REVERSE # OF SEEKS	REVERSE SEEK TIME

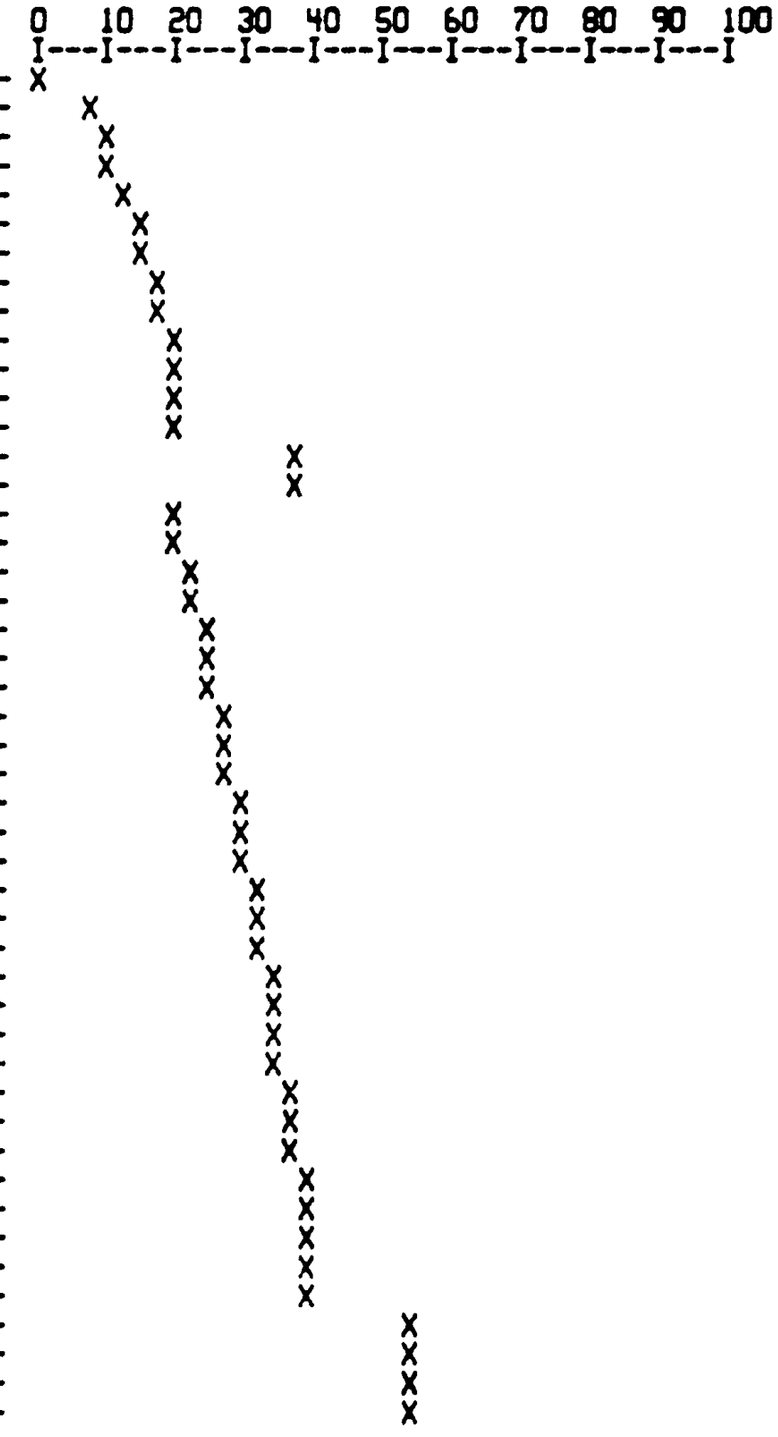
```
                                                    200   45000     156   45000
                                                                    44   60000
```

H02

8.3.2 SEEK GRAPH TEST (TEST #12)

EXAMPLE: SMALL SEEK GRAPH (SWITCH (04) = 0)

SEEK TIME GRAPH
X AXIS - SEEK TIME - MILLI SECS
Y AXIS - CYLINDER SEEKED FROM 0



```
195-                         X
200-                 X
202-                 X
```

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.

8.5 DEFAULT ADDRESSES, TEST SEQUENCE, AND PARAMETERS

8.5.1 DEFAULT ADDRESS

THE PROGRAM IS LOADED FOR THE FOLLOWING ADDRESSES:

RP11E BUS ADDRESS - 176710
RP11E VECTOR ADDRESS - 254
RP11E PRIORITY - 5

TO ALTER THESE ADDRESS, START THE PROGRAM AT LOCATION 210 OR 214.

8.5.2 DEFAULT TEST SEQUENCE

THE DEFAULT TEST SEQUENCE IS TESTS 0 - 7. THIS SEQUENCE IS RESTORED ANY TIME A START FROM 200, 204, 210, OR 214 IS INITIATED. 'CONTROL C' RESTART DOES NOT ALTER THE TEST SEQUENCE.

8.5.3 DEFAULT TEST PARAMETERS

THE DEFAULT TEST PARAMETERS ARE GIVEN BELOW. REFER TO SECTION 4.3.1 FOR A GLOSSARY OF THE MNEMONICS USED TO IDENTIFY THE PARAMETERS. THE DEFAULT TEST PARAMETERS ARE RESTORED WHEN A START AT 200, 204, 210, OR 214 IS INITIATED. THE 'CONTROL C' RESTART DOES NOT ALTER THE CURRENT PARAMETER VALUES.

		PARAMETERS			
TEST #	'R'	'FC'	'LC'	'IC'	'TK'
TEST 0	100	-	-	-	0
TEST 1	1000	0	MAXCYL	-	0
TEST 2	10	0	MAXCYL	1	0
TEST 3	10	0	NOTE 3	1	0
TEST 4	10	0	MAXCYL	1	0
TEST 5	10	0	MAXCYL	1	0
TEST 6	10	0	MAXCYL	1	0
TEST 7	1	0	MAXCYL	1	0
TEST 10	(SEE BELOW - NOTE 1)				
TEST 11	1	0	MAXCYL	-	-

```
                                        TEST 12     (SEE BELOW - NOTE 2)
```

NOTE 1: TEST 10 ASSUMES THE FOLLOWING DEFAULT PARAMETERS. THESE PARAMETERS CANNOT BE ALTERED BY THE OPERATOR.

1	0	MAXCYL	1	-

NOTE 2: TEST 12 ASSUMES THE FOLLOWING DEFAULT PARAMETERS. THESE PARAMETERS CANNOT BE ALTERED BY THE OPERATOR.

1	0	MAXCYL	1	-

NOTE 3: 'MAXCYL' USED ABOVE IS USED TO DENOTE THE MAXIMUM CYLINDER ADDRESS USED BY THE PROGRAM. THE VALUE OF 'MAXCYL' IS DETERMINED BY THE TYPE OF DRIVES USED ON THE SYSTEM. 'MAXCYL' IS 405 (10) IF THE DRIVES ARE RP03'S AND IS 202 (10) IF THE DRIVES ARE RP02'S. IN TEST 3, 'LC' WILL BE 128 (10) IF THE DRIVES ARE RP02'S AND 256 (10) IF THE DRIVES ARE RP03'S.

8.6 TIMING TEST RESTRICTIONS

THE TIMING TESTS (TESTS 10, 11, & 12) ARE NOT INTENDED TO BE 'PASS-FAIL' TESTS. BECAUSE A HIGH RESOLUTION CLOCK IS NOT USED, THE TIMES ARE APPROXIMATE AND ARE INTENDED TO BE A GENERAL INDICATION OF THE CONDITION OF THE DRIVE'S SERVO SYSTEM. NO ATTEMPT SHOULD BE MADE TO ADJUST THE SERVO SYSTEM USING THE OUTPUT FROM THESE TESTS.

8.7 POSITIONING CHECK

IF SW<00> IS NOT SET, THE PROGRAM WILL CHECK THE DISK'S POSITION AFTER EACH SEEK (EXCEPT IN TESTS 10, 11, & 12).

THE PROGRAM CHECKS POSITION IN THE FOLLOWING MANNER:

USING THE 'SOT' COUNTER TO DETERMINE THE ADDRESS, THE PROGRAM READS THE HEADER FROM THE NEXT SECTOR AND COMPARES THE CYLINDER ADDRESS IN THE HEADER AGAINST THE CYLINDER THE PROGRAM EXPECTED; IF THE CYLINDER ADDRESS IN THE HEADER DOESN'T AGREE WITH THE EXPECTED CYLINDER ADDRESS, THE PROGRAM REPORTS A POSITIONING ERROR AND ISSUES A HOME SEEK TO THE DRIVE. (THE PROGRAM ALSO CHECKS POSITION AFTER A HOME SEEK OPERATION.)

9. PROGRAM DESCRIPTION

THE RP11E DRIVE POSITIONING TEST CONTAINS 11 TESTS. TESTS 0 - 6 TEST THE OPERATION OF THE DRIVE'S SERVO SYSTEM. POSITIONING ACCURACY IS CHECKED BY READING A HEADER FROM THE DESTINATION CYLINDER USING A 'READ WITHOUT IMPLIED SEEK' INSTRUCTION AFTER THE SEEK TERMINATES.

TESTS 10, & 11, 12 MEASURE SEEK TIME. TEST 10 & 11 MEASURE THE TIME FOR SPECIFIC SEEK OPERATIONS; TEST 12 MEASURES THE TIME REQUIRED TO SEEK FROM CYLINDER 0 TO ALL OTHER CYLINDERS. TESTS 10 & 11 TYPE OUT THE MEASURED SEEK TIME IN MICROSECONDS WHILE TEST 12 PRODUCES A SEEK TIME GRAPH. IN THE SEEK TIMING TESTS, POSITIONING ACCURACY IS NOT CHECKED.

SEEK TIME IN TESTS 10, 11, & 12 IS DETERMINED BY COUNTING THE NUMBER OF SECTOR PULSES OCCURING BETWEEN START OF SEEK AND SEEK DONE. THE 'SOT' COUNTER IS USED AS THE REFERENCE IN THESE TESTS. THE RESOLUTION OF THE TIMING TESTS IS 2.5 MS.

THE PROGRAM WILL START BY IDENTIFING ITSELF AND DETERMINING ALL DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH THE LOWEST NUMBERICAL DRIVE AND PROCEEDING IN SEQUENCIAL ORDER ALL OF THE DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST SEQUENCE (TESTS 0 - 7) WILL BE PERFORMED ON EACH DRIVE BEFORE MOVING TO THE NEXT DRIVE IN SEQUENCE. THE 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 SECTION FOR A DETAILED DESCRIPTION OF EACH TEST.

9.1 TEST DESCRIPTIONS

IN THE DESCRIPTIONS OF THE FOLLOWING TESTS, THE VARIABLES USED AND THEIR DEFAULT VALUES (UNLESS SPECIFIED OTHERWISE) ARE:

MNEMONIC	VARIABLE
FC	FIRST CYLINDER ADDRESS
LC	LAST CYLINDER ADDRESS
IC	INCREMENT VALUE
NC OR NC1	NEW OR MODIFIED CYLINDER (FC+IC) ADDRRESS
NC2	NEW OR MODIFIED CYLINDER (LC-IC) ADDRESS
TK	TRACK ADDRESS

```
                        9.1.1   TEST 0 - HOME SEEK TEST

                                THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A HOME SEEK
                                COMMAND CYCLE.  AT THE COMPLETION OF THE COMMAND, THE
                                STATUS INDICATORS ARE CHECKED TO VERIFY THAT NO ERRORS
                                HAVE OCCURED.

                        9.1.2   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 INSURE PROPER OPERATION.

                        9.1.3   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
                                INSURE PROPER OPERATION.

                        9.1.4   TEST 3 - STEPPING SEEK TEST

                                THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4,
                                8, 16, 32, 64, AND 128 (AND 256 IF AN RP03).  AT THE
                                COMPLETION OF EACH SEEK COMMAND, THE PROPER INDICATORS ARE
                                EXAMINED TO VERIFY PROPER OPERATION.

                        9.1.5   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 VERIFY PROPER OPERATION.

                        9.1.6   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
                                VERIFY PROPER OPERATION.  "NC1" AND "NC2" DEFAULT TO
                                "FC" AND "LC" RESPECTIVELY.

                        9.1.7   TEST 6 - SERVO ADDRESSING LOGIC NOSE GENERATION

                                IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
                                NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5.  NOW "NC" IS UPDATED
                                BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNITL "LC" IS
```

```
960                                     EXCEEDED BY ANY OF THE ABOVE VALUES.  THE INITIAL VALUE OF "NC"
961                                     IS "FC".  AT THE COMPLETION OF EACH SEEK COMMAND THE
962                                     PROPER INDICATORS ARE EXAMINED TO INSURE PROPER OPERATION.
963
```

964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010

9.1.8 TEST 7 - CYLINDER ADDRESSING TEST

THIS TEST WILL CAUSE THE DRIVE TO SEEK FROM EACH CYLINDER TO EVERY OTHER CYLINDER BETWEEN CYLINDERS "FC" AND "LC". SEEK CYCLES ARE COMMANDED FROM CYLINDER "NC" TO CYLINDER "NC1" AND BACK TO CYLINER "NC". "NC" AND "NC1" START AT "FC". "NC1" IS INCREMENTED BY "IC" UNTIL "NC1" IS EQUAL TO "LC"; WHEN "NC1" IS EQUAL TO "LC", IT IS RESET TO "FC" AND "NC" IS INCREMENTED BY "IC". THE CYCLE IS COMPLETE WHEN "NC" AND "NC1" BOTH EQUAL "LC". AT THE COMPLETION OF EACH SEEK COMMAND, THE INDICATORS ARE EXAMINED TO VERIFY PROPER OPERATION.

9.1.9 TEST 10 - ONE CYLINDER SEEK TIMER TEST

THIS TEST PERFORMS ONE CYLINDER SEEKS FROM CYLINDER 0 TO THE MAXIMUM CYLINDER FOR THE DRIVE (E.G. 202 FOR RP02'S AND 405 FOR RP03'S). THE SEEK TIMES ARE MEASURED RELATIVE TO THE SECTOR PULSES FROM THE DRIVE. THE TEST DISPLAYS THE FORWARD AND REVERSE TIMES.

9.1.10 TEST 11 - SEEK-SEEK TIMER TEST

THIS TEST IS A GENERAL PURPOSE TIMING TEST WHICH PERFORMS 200 ITERATIONS BETWEEN THE CYLINDER IN 'FC' AND THE CYLINDER IN 'LC'. BOTH FORWARD AND REVERSE SEEKS ARE TIMED AND DISPLAYED. THE SEEK TIMES DISPLAYED ARE RELATIVE TO THE SECTOR COUNTER. 'FC' DEFAULTS TO 0; 'LC' DEFAULTS TO 'MAXCYL' (202 IF RP02'S OR 405 IF RP03'S).

9.1.11 TEST 12 - SEEK TIME GRAPH TEST

THIS TEST PRODUCES TWO SEEK TIME GRAPHS. THE FIRST GRAPH IS TYPED IS SWITCH <04> IS 0. THIS GRAPH IS A PLOT OF SEEK TIME AGAINST CYLINDERS SEEKED TO FROM CYLINDER 0. THE CYLINDERS PLOTTED ARE: 0,1,2,3,4,6,8,10,12,14,16,18,20,25...100,105...MAXCYL. THE OTHER GRAPH WHICH IS PRODUCED (SWITCH <04> SET) IS TIME TO SEEK TO ALL CYLINDERS FROM CYLINDER 0.

IN BOTH SEEK GRAPHS, ONLY THE FORWARD SEEK TIMES ARE MEASURED.

THIS TEST USES THE SECTOR COUNTER ('SOT') IN THE RF11 AS THE TIME BASE FOR THE MEASURMENTS, USE OF THE 'SOT' PROVIDES A RESOLUTION OF 2.5 MS.

10. PROGRAM LISTING

```
                                        %
                                .TITLE  MD-11-DZRPZ-B, RP11E DRIVE POSITIONING TEST
                                ;*COPYRIGHT (C) 1975,1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY C. HESS/F. ROEMER
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          ------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         6            TYPE ALL THE RP11 REGISTERS IF ERROR
                                ;*         4            TYPE THE LONG SEEK TIME GRAPH
                                ;*         3            PERFORM STALL BETWEEN SEEK ORDERS
                                ;*         2            USE RANDOM STALL VALUE BETWEEN SEEK ORDERS
                                ;*         1            PERFORM INCREMENTING STALL IN TEST 4
                                ;*         0            DO NOT CHECK POSITION AFTER SEEK ORDERS

                                .SBTTL  TRAP CATCHER

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

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000200                          $SVPC=.                 ;SAVE PC
        000046                          .=46
000046  007712                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
        000200                          .=$SVPC                 ;; RESTORE PC

                                .SBTTL  STARTING ADDRESSES
        000200                  .=200
                                ;*200 = NORMAL START
000200  000137  002246                  JMP     @#START1
```

```
1067                                    ;*204 = SELECT OPERATING PARAMETERS
1068  000204  000137  002270                    JMP     @#START2
1069                                    ;*210 = SELECT RP11 ADDRESSES
1070  000210  000137  002236                    JMP     @#START3
1071                                    ;*214 = COMBINATION OF 204 AND 210
1072  000214  000137  002260                    JMP     @#START4
1073
1074                                    .SBTTL  BASIC DEFINITIONS
1075
1076                                    ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1077          001100                    STACK=  1100
1078                                    .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
1079                                    .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
1080
1081                                    ;*MISCELLANEOUS DEFINITIONS
1082          000011                    HT=     11              ;;CODE FOR HORIZONTAL TAB
1083          000012                    LF=     12              ;;CODE FOR LINE FEED
1084          000015                    CR=     15              ;;CODE FOR CARRIAGE RETURN
1085          000200                    CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
1086          177776                    PS=     177776          ;;PROCESSOR STATUS WORD
1087                                    .EQUIV  PS,PSW
1088          177774                    STKLMT= 177774          ;;STACK LIMIT REGISTER
1089          177772                    PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
1090          177570                    DSWR=   177570          ;;HARDWARE SWITCH REGISTER
1091          177570                    DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
1092
1093                                    ;*GENERAL PURPOSE REGISTER DEFINITIONS
1094          000000                    R0=     %0              ;;GENERAL REGISTER
1095          000001                    R1=     %1              ;;GENERAL REGISTER
1096          000002                    R2=     %2              ;;GENERAL REGISTER
1097          000003                    R3=     %3              ;;GENERAL REGISTER
1098          000004                    R4=     %4              ;;GENERAL REGISTER
1099          000005                    R5=     %5              ;;GENERAL REGISTER
1100          000006                    R6=     %6              ;;GENERAL REGISTER
1101          000007                    R7=     %7              ;;GENERAL REGISTER
1102                                    .EQUIV  R6,SP           ;;STACK POINTER
1103                                    .EQUIV  R7,PC           ;;PROGRAM COUNTER
1104
1105                                    ;*PRIORITY LEVEL DEFINITIONS
1106          000000                    PR0=    0               ;;PRIORITY LEVEL 0
1107          000040                    PR1=    40              ;;PRIORITY LEVEL 1
1108          000100                    PR2=    100             ;;PRIORITY LEVEL 2
1109          000140                    PR3=    140             ;;PRIORITY LEVEL 3
1110          000200                    PR4=    200             ;;PRIORITY LEVEL 4
1111          000240                    PR5=    240             ;;PRIORITY LEVEL 5
1112          000300                    PR6=    300             ;;PRIORITY LEVEL 6
1113          000340                    PR7=    340             ;;PRIORITY LEVEL 7
1114
1115                                    ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1116          100000                    SW15=   100000
1117          040000                    SW14=   40000
1118          020000                    SW13=   20000
1119          010000                    SW12=   10000
1120          004000                    SW11=   4000
1121          002000                    SW10=   2000
1122          001000                    SW09=   1000
```

```
                000400          SW08=   400
                000200          SW07=   200
                000100          SW06=   100
                000040          SW05=   40
                000020          SW04=   20
                000010          SW03=   10
                000004          SW02=   4
                000002          SW01=   2
                000001          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                100000          BIT15=  100000
                040000          BIT14=  40000
                020000          BIT13=  20000
                010000          BIT12=  10000
                004000          BIT11=  4000
                002000          BIT10=  2000
                001000          BIT09=  1000
                000400          BIT08=  400
                000200          BIT07=  200
                000100          BIT06=  100
                000040          BIT05=  40
                000020          BIT04=  20
                000010          BIT03=  10
                000004          BIT02=  4
                000002          BIT01=  2
                000001          BIT00=  1
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014          TBITVEC=14              ;;"T" BIT
                000014          TRTVEC= 14              ;;TRACE TRAP
                000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024          PWRVEC= 24              ;;POWER FAIL
```

```
1179    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1180    000034          TRAPVEC=34              ;;"TRAP" TRAP
1181    000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
1182    000064          TPVEC=  64              ;;TTY PRINTER VECTOR
1183    000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1184
1185                    ;OP CODE DEFINITIONS
1186
1187    000001          CLEAR=1                         ;CONTROLLER CLEAR
1188    000003          WRTSEK=3                        ;WRITE WITH IMPLIED SEEK AND HEAD SELECTION
1189    000005          RDSEK=5                         ;READ WITH IMPLIED SEEK AND HEAD SELECT
1190    000007          WCKSEK=7                        ;WRITE CHECK WITH IMPLIIED SEEK AND HEAD SELECT
1191    000011          SEEK=11                         ;SEEK
1192    000013          WRITE=13                        ;WRITE (NO IMPLIED SEEK OR HEAD SELECT)
1193    000015          HOMSEK=15                       ;HOME SEEK
1194    000017          READ=17                         ;READ (NO IMPLIED SEEK OR HEAD SELECT)
1195
1196                    ;RP11 REGISTER EQUATES
1197
1198                    ;RPDS (DRIVE STATUS REGISTER)
1199
1200                    ;ATTENTION BITS ARE REFERENCED BY BIT NUMBER (BIT00 - BIT07)
1201    000400          SWP=400                         ;SELECTED UNIT WRITE PROTECTED
1202    001000          SUFU=1000                       ;SELECTED UNIT FILE UNSAFE
1203    002000          SUSU=2000                       ;SELECTED UNIT SEEK UNDERWAY
1204    004000          SUSI=4000                       ;SELECTED UNIT SEEK INCOMPLETE
1205    010000          HNF=10000                       ;HEADER NOT FOUND
1206    020000          SURP03=20000                    ;SELECTED UNIT IS AN RP03
1207    040000          SUOL=40000                      ;SELECTED UNIT IS ONLINE
1208    100000          SURDY=100000                    ;SELECTED UNIT IS READY
1209
1210                    ;RPER (ERROR REGISTER)
1211
1212    000001          DSKERR=1                        ;DISK ERROR
1213    000002          EOP=2                           ;END OF PACK
1214    000004          NXME=4                          ;NON-EXISTENT MEMORY
1215    000010          WCE=10                          ;WRITE CHECK ERROR
1216    000020          TIMEE=20                        ;TIMING ERROR
1217    000040          CSME=40                         ;CHECKSUM ERROR
1218    000100          WPE=100                         ;WORD PARITY ERROR
1219    000200          LPE=200                         ;LONGITUDINAL PARITY ERROR
1220    000400          MODERR=400                      ;MODE ERROR
1221    001000          FMTE=1000                       ;FORMAT ERROR
1222    002000          PROG=2000                       ;PROGRAM ERROR
1223    004000          NXS=4000                        ;NON-EXISTENT SECTOR
1224    010000          NXT=10000                       ;NON-EXISTENT TRACK
1225    020000          NXC=20000                       ;NON-EXSITENT CYLINDER
1226    040000          FUV=40000                       ;FILE UNSAFE VIOLATION
1227    100000          WPV=100000                      ;WRITE PROTECT VIOLATION
1228
1229                    ;RPCS  (CONTROL REGISTER)
1230
1231                    ;'GO' BIT AND FUNCTION BITS ARE LOADED TOGETHER AND ARE DEFINED ELSEWHERE.
1232                    ;EXTENDED MEMORY ADDRESS BITS ARE REFERED TO BY BIT NUMBER
1233    000100          IDE=100                         ;INTERRUPT ON DONE
1234    000200          RDY=200                         ;RP11 READY
```

```
1235                                    ;DRIVE SELECT BITS ARE REFERED TO BY BIT NUMBER
1236            004000                  HDR=4000                        ;HEADER
1237            010000                  MODE=10000                      ;MODE
1238            020000                  AIE=20000                       ;ATTENTION INTERRUPT ENABLE
1239            040000                  HE=40000                        ;HARD ERROR
1240            100000                  ERR=100000                      ;ERROR
1241
```

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102      000         $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103      000         $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114      000         $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115      001         $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134      000         $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135      000         $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          $SWR:   .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154      000         $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155      002         $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156      012         $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157      000         $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000          $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000          $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000          $REG10: .WORD   0               ;;CONTAINS (($REGAD)+20)
001204  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001206  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001210  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001212  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001216      077         $QUES:  .ASCII  /?/             ;;QUESTION MARK
001217      015         $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001220  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;**********************************************************************
```

```
1298  001222  000000                CNTRLC: .WORD   0               ;CONTROL "C" FLAG
1299  001224  000000                BUSADR: .WORD   0               ;GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
1300  001226  000000                INADR:  .WORD   0               ;CONTAINS INNER ADDRESS FOR TIMING TEST
1301  001230  000000                OUTADR: .WORD   0               ;CONTAINS OUTER ADDRESS FOR TIMING TEST
1302  001232  000000                DRVSEL: .WORD   0               ;DRIVES SELECTED FOR TESTING
1303  001234  000777                TSTNMS: .WORD   777             ;RUN TESTS 0-10
1304  001236  000000                MAXCYL: .WORD   0               ;MAXIMUM CYLINDER ADDRESS
1305  001240  000000                DRVTYP: .WORD   0               ;CONTAINS A BIT FOR EACH RP03 DRIVE
1306                                                                ;ON THE SYSTEM.  BIT00=DRIVE 0, ETC.
1307  001242  000000                SPSAV:  .WORD   0               ;SAVE THE STACK POINTER HERE
1308  001244  000000                BYPASS: .WORD   0               ;BYPASS CURRENT TEST ADDRESS
1309  001246  000000                CHKDRV: .WORD   0               ;DRIVE UNDER TEST
1310  001250  000000                DRVMSK: .WORD   0               ;DRIVE MASK BIT
1311  001252  000000                DRVBAD: .WORD   0               ;CONTAINS BITS FOR UNSAFE OR OFFLINE DRIVES
1312  001254  000000                CYL.CR: .WORD   0               ;CURRENT CYLINDER
1313  001256  000000                TRK.RD: .WORD   0               ;TRACK READ
1314  001260  000000                SEC.RD: .WORD   0               ;SECTOR READ
1315  001262  000000                CYL.DS: .WORD   0               ;CYLINDER DESIRED
1316  001264  000000                STR20X: .WORD   0               ;START FROM 200, 204, 210, 214 FLAG
1317
1318                                ;CONSTANTS STORAGE
1319  001266  176710                RPADR:  .WORD   176710          ;RP11 ADDRESS
1320  001270  000254                RPVEC:  .WORD   254             ;RP11 VECTOR ADDRESS
1321  001272  000240                RPPRIO: .WORD   5*32.           ;RP11 PRIORITY
1322  001274  000000                STALL0: .WORD   0               ;VARIABLE STALL (TEST 4)
1323  001276  000031                STALL1: .WORD   25.             ;25 MILLISECONDS STALL (TEST 0-6)
1324  001300  000000                STALLG: .WORD   0               ;GENERAL STALL VALUE (VARIABLE)
1325  001302  000062                MXSTAL: .WORD   50.             ;MAX INCREMENTING STALL ALLOWED IN TEST 4
1326                                        .EVEN
1327
1328                                ;*TABLE OF DRIVE STATUS INDICATORS
1329                                        ;*DRVSTA>0 - DRIVE IS ONLINE
1330                                        ;*DRVSTA<0 - DRIVE IS OFFLINE, UNSAFE, OR AN RP03
1331
1332  001304  000                   DRVSTA: .BYTE   0               ;DRIVE 0
1333  001305  000                           .BYTE   0               ;DRIVE 1
1334  001306  000                           .BYTE   0               ;DRIVE 2
1335  001307  000                           .BYTE   0               ;DRIVE 3
1336  001310  000                           .BYTE   0               ;DRIVE 4
1337  001311  000                           .BYTE   0               ;DRIVE 5
1338  001312  000                           .BYTE   0               ;DRIVE 6
1339  001313  000                           .BYTE   0               ;DRIVE 7
1340
1341                                ;ATTENTION BIT TABLE
1342
1343  001314  001                   ATABIT: .BYTE   1               ;DRIVE 0
1344  001315  002                           .BYTE   2               ;DRIVE 1
1345  001316  004                           .BYTE   4               ;DRIVE 2
1346  001317  010                           .BYTE   10              ;DRIVE 3
1347  001320  020                           .BYTE   20              ;DRIVE 4
1348  001321  040                           .BYTE   40              ;DRIVE 5
1349  001322  100                           .BYTE   100             ;DRIVE 6
1350  001323  200                           .BYTE   200             ;DRIVE 7
1351
1352                                ;BIT TABLE
1353  001324  000001                BITS:   .WORD   BIT00
```

```
001326  000002                          .WORD   BIT01
001330  000004                          .WORD   BIT02
001332  000010                          .WORD   BIT03
001334  000020                          .WORD   BIT04
001336  000040                          .WORD   BIT05
001340  000100                          .WORD   BIT06
001342  000200                          .WORD   BIT07
001344  000400                          .WORD   BIT08
001346  001000                          .WORD   BIT09
001350  002000                          .WORD   BIT10
001352  004000                          .WORD   BIT11
001354  010000                          .WORD   BIT12
001356  020000                          .WORD   BIT13
001360  040000                          .WORD   BIT14
001362  100000                          .WORD   BIT15

                        ;DRIVE OPERATION PARAMETER BLOCK

001364     000  DPB:     .BYTE   0               ;OP CODE
001365     000          .BYTE   0               ;DRIVE ADDRESS
001366  000000          .WORD   0               ;CYLINDER ADDRESS
001370     000          .BYTE   0               ;SECTOR ADDRESS
001371     000          .BYTE   0               ;TRACK ADDRESS

                        ;COMMON STORAGE FOR TEST PARAMETER
001372  000000  PRM:     .WORD   0
001374  000000  PPT:     .WORD   0               ;REPEAT COUNTS FOR ALL TESTS
001376  000000  FC:      .WORD   0               ;FIRST CYLINDER
001400  000000  LC:      .WORD   0               ;LAST CYLINDER
001402  000000  IC:      .WORD   0               ;INCREMENT CYLINDER
001404  000000  TK:      .WORD   0               ;TRACK ADDRESS

                        ;TABLE OF PARAMETER POINTERS
001406  001632  PRMPT:   .WORD   PRM0
001410  001640          .WORD   PRM1
001412  001654          .WORD   PRM2
001414  001670          .WORD   PRM3
001416  001704          .WORD   PRM4
001420  001720          .WORD   PRM5
001422  001734          .WORD   PRM6
001424  001750          .WORD   PRM7
001426  001764          .WORD   PRM10
001430  001770          .WORD   PRM11
001432  002000          .WORD   PRM12

                        ;PARAMETER UPPER LIMIT
001434  077777  PRMLMT:  .WORD   32767.          ;"R"
001436  000312          .WORD   202.            ;"FC"
001440  000312          .WORD   202.            ;"LC"
001442  000312          .WORD   202.            ;"IC"
001444  000023          .WORD   19.             ;"TK"

                        ;TABLE OF MESSAGE POINTERS
001446  021370  PRMMSG:  .WORD   MSG.R
001450  021372          .WORD   MSG.FC
```

```
1410  001452  021375                          .WORD   MSG.LC
1411  001454  021400                          .WORD   MSG.IC
1412  001456  021403                          .WORD   MSG.TK
1413
1414                                  ;DEFAULT VALUES OF TESTS PARAMETERS
1415  001460  000021  000144  000000  DFLT:   .WORD   21,100.,0          ;HOME SEEK (T0)
1416  001466  000037  001750  000000          .WORD   37,1000.,0,202.,0,0  ;SEEK/SEEK (T1)
1417  001474  000312  000000  000000
1418  001502  000037  000010  000000          .WORD   37,10,0,202.,1,0  ;INCREMENT SEEK (T2)
1419  001510  000312  000001  000000
1420  001516  000037  000010  000000          .WORD   37,10,0,128.,1,0  ;STEPPING SEEK (T3)
1421  001524  000200  000001  000000
1422  001532  000037  000010  000000          .WORD   37,10,0,202.,1,0  ;OSCILLATING SEEK (T4)
1423  001540  000312  000001  000000
1424  001546  000037  000010  000000          .WORD   37,10,0,202.,1,0  ;CONVERGING/DIVERGING SEEK (T5)
1425  001554  000312  000001  000000
1426  001562  000037  000010  000000          .WORD   37,10,0,202.,1,0  ;SERVO ADDRESSING LOGIC NOISE (T6)
1427  001570  000312  000001  000000
1428  001576  000037  000001  000000          .WORD   37,1,0,202.,1,0  ;CYLINDER ADDRESSING TEST (T7)
1429  001604  000312  000001  000000
1430  001612  000001  000001                  .WORD   1,1              ;ONE CYLINDER SEEK TIMER TEST (T10)
1431  001616  000007  000001  000000          .WORD   7,1,0,202.       ;SEEK-SEEK TIMER TEST (T11)
1432  001624  000312
1433  001626  000001  000001                  .WORD   1,1              ;SEEK TIME GRAPH TEST (T12)
1434
1435                                  ;PARAMETER TABLES
1436
1437                                  ;HOME SEEK (T0)
1438  001632  000021                  PRM0:   .WORD   21
1439  001634  000144                  RPT0:   .WORD   100.
1440  001636  000000                  TK0:    .WORD   0
1441
1442                                  ;SEEK-SEEK (T1)
1443  001640  000037                  PRM1:   .WORD   37
1444  001642  001750                  RPT1:   .WORD   1000.
1445  001644  000000                  FC1:    .WORD   0
1446  001646  000312                  LC1:    .WORD   202.
1447  001650  000000                  IC1:    .WORD   0
1448  001652  000000                  TK1:    .WORD   0
1449
1450                                  ;INCREMENT SEEK (T2)
1451  001654  000037                  PRM2:   .WORD   37
1452  001656  000012                  RPT2:   .WORD   10.
1453  001660  000000                  FC2:    .WORD   0
1454  001662  000312                  LC2:    .WORD   202.
1455  001664  000001                  IC2:    .WORD   1
1456  001666  000000                  TK2:    .WORD   0
1457
1458                                  ;STEPPING SEEK (T3)
1459  001670  000037                  PRM3:   .WORD   37
1460  001672  000012                  RPT3:   .WORD   10.
1461  001674  000000                  FC3:    .WORD   0
1462  001676  000200                  LC3:    .WORD   128.
1463  001700  000001                  IC3:    .WORD   1
1464  001702  000000                  TK3:    .WORD   0
1465
```

```
1466                                    ;OSCILLATING SEEK (T4)
1467  001704  000037                    PRM4:   .WORD   37
1468  001706  000012                    RPT4:   .WORD   10.
1469  001710  000000                    FC4:    .WORD   0
1470  001712  000312                    LC4:    .WORD   202.
1471  001714  000001                    IC4:    .WORD   1
1472  001716  000000                    TK4:    .WORD   0
1473
1474                                    ;CONVERGING/DIVERGING SEEK (T5)
1475  001720  000037                    PRM5:   .WORD   37
1476  001722  000012                    RPT5:   .WORD   10.
1477  001724  000000                    FC5:    .WORD   0
1478  001726  000312                    LC5:    .WORD   202.
1479  001730  000001                    IC5:    .WORD   1
1480  001732  000000                    TK5:    .WORD   0
1481
1482                                    ;SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)
1483  001734  000037                    PRM6:   .WORD   37
1484  001736  000012                    RPT6:   .WORD   10.
1485  001740  000000                    FC6:    .WORD   0
1486  001742  000312                    LC6:    .WORD   202.
1487  001744  000001                    IC6:    .WORD   1
1488  001746  000000                    TK6:    .WORD   0
1489
1490                                    ;CYLINDER ADDRESSING (T7)
1491  001750  000037                    PRM7:   .WORD   37
1492  001752  000001                    RPT7:   .WORD   1
1493  001754  000000                    FC7:    .WORD   0
1494  001756  000312                    LC7:    .WORD   202.
1495  001760  000001                    IC7:    .WORD   1
1496  001762  000000                    TK7:    .WORD   0
1497
1498                                    ;ONE CYLINDER SEEK TIMER TEST (T10)
1499  001764  000001                    PRM10:  .WORD   1
1500  001766  000001                    RPT10:  .WORD   1
1501
1502                                    ;SEEK-SEEK TIMING TEST (T11)
1503  001770  000007                    PRM11:  .WORD   7
1504  001772  000001                    RPT11:  .WORD   1
1505  001774  000000                    FC11:   .WORD   0
1506  001776  000312                    LC11:   .WORD   202.
1507
1508                                    ;SEEK TIME GRAPH TEST (T12)
1509  002000  000001                    PRM12:  .WORD   1
1510  002002  000001                    RPT12:  .WORD   1
1511
1512                                    ;SAVE THE RP11E REGISTERS HERE
1513
1514  002004  000000                    $RPDS:  .WORD   0               ;DRIVE STATUS REGISTER
1515  002006  000000                    $RPER:  .WORD   0               ;ERROR REGISTER
1516  002010  000000                    $RPCS:  .WORD   0               ;COMMAND & STATUS REGISTER
1517  002012  000000                    $RPWC:  .WORD   0               ;WORD COUNT REGISTER
1518  002014  000000                    $RPBA:  .WORD   0               ;BUFFER ADDRESS REGISTER
1519  002016  000000                    $RPCA:  .WORD   0               ;CURRENT CYLINDER ADDRESS REGISTER
1520  002020  000000                    $RPDA:  .WORD   0               ;TRACK-SECTOR ADDRESS REGISTER
1521  002022  000000                    $RPM1:  .WORD   0               ;MAINTENANCE REGISTER #1
```

```
1522  002024  000000                  $SUCA:  .WORD   0               ;SELECTED UNIT CYLINDER ADDRESS REGISTER
1523  002026  000000                  $SILO:  .WORD   0               ;SILO REGISTER
1524
1525                                  ;REGISTER INDEX VALUES
1526
1527          000000                  RPDS=00                         ;DRIVE STATUS REGISTER
1528          000002                  RPER=02                         ;ERROR REGISTER
1529          000004                  RPCS=04                         ;CONTROL REGISTER
1530          000006                  RPWC=06                         ;WORD COUNT REGISTER
1531          000010                  RPBA=10                         ;BUFFER ADDRESS REGISTER
1532          000012                  RPCA=12                         ;CYLINDER ADDRESS REGISTER
1533          000014                  RPDA=14                         ;SECTOR/TRACK ADDRESS REGISTER
1534          000016                  RPM1=16                         ;MAINTENANCE REGISTER #1
1535          000020                  RPM2=20                         ;MAINTENANCE REGISTER #2
1536          000022                  RPM3=22                         ;MAINTENANCE REGISTER #3
1537          000024                  SUCA=24                         ;SELECTED UNIT CYLINDER ADDRESS REGISTER
1538          000026                  SILO=26                         ;SILO REGISTER
1539
1540
1541  002030  020040      040         BLNK13: .ASCII  /   /
1542  002033      040                 BLNK10: .ASCII  / /
1543  002034      040                 BLNKS9: .ASCII  / /
1544  002035      040                 BLNKS8: .ASCII  / /
1545  002036      040                 BLNKS7: .ASCII  / /
1546  002037      040                 BLNKS6: .ASCII  / /
1547  002040      040                 BLNKS5: .ASCII  / /
1548  002041      040                 BLNKS4: .ASCII  / /
1549  002042      040                 BLNKS3: .ASCII  / /
1550  002043      040                 BLNKS2: .ASCII  / /
1551  002044  000040                  BLNKS1: .ASCIZ  / /
1552
1553                                          .EVEN
```

```
                                .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

002046                          $ERRTB:
                                ;*EM AND DH ARE ASCIZ MESSAGES, DT IS A STRING OF WORDS THAT POINT TO THE
                                ;*DATA TO BE TYPED AND DF IS A STRING OF DATA THAT TELL HOW THE DT WORDS
                                ;*ARE TO BE TYPED.  IF ANY OF THE POINTERS ARE NOT NEEDED FOR A PARTICULAR
                                ;*ERROR IT IS REPLACED WITH A ZERO.
                                ;*EACH OF THE ITEMS BELOW REFER TO THE ERROR NUMBER AND INDICATE
                                ;*THE INFORMATION THAT WILL BE TYPED WHEN THE ERROR OCCURS.
                                ;*UNLESS STATED OTHER ALL NUMBERS ARE OCTAL


                                ;ERROR 1
002046  021612                          EM1             ;RP11 FAILED TO RESPOND TO ADDRESSING
002050  022553                          DH1
002052  023306                          DT1
002054  023442                          DF1

                                ;ERROR 2
002056  021657                          EM2             ;DRIVE UNSAFE
002060  022570                          DH2
002062  023312                          DT2
002064  023446                          DF2

                                ;ERROR 3
002066  021674                          EM3             ;DRIVE OFFLINE
002070  022570                          DH2
002072  023312                          DT2
002074  023446                          DF2

                                ;ERROR 4
002076  021712                          EM4             ;DISK ERROR DURING TIMING TEST
002100  022570                          DH2
002102  023312                          DT2
002104  023446                          DF2

                                ;ERROR 5
002106  021751                          EM5             ;NO INTERRUPT FROM POSITIONING AFTER 1 SECOND
```

```
1610  002110  022570                    DH2
1611  002112  023312                    DT2
1612  002114  023446                    DF2
1613
1614                            ;ERROR 6
1615
1616  002116  022026                    EM6             ;DISK ERROR AFTER POSITIONING
1617  002120  022570                    DH2
1618  002122  023312                    DT2
1619  002124  023446                    DF2
1620
1621                            ;ERROR 7
1622
1623  002126  022063                    EM7             ;ATTN BIT NOT SET AFTER POSITIONING
1624  002130  022570                    DH2
1625  002132  023312                    DT2
1626  002134  023446                    DF2
1627
1628                            ;ERROR 10
1629
1630  002136  022136                    EM10            ;DRIVE OFFLINE AFTER POSITIONING
1631  002140  022570                    DH2
1632  002142  023312                    DT2
1633  002144  023446                    DF2
1634
1635                            ;ERROR 11
1636
1637  002146  022176                    EM11            ;'SUCA' NOT CORRECT AFTER POSITIONING
1638  002150  022645                    DH11
1639  002152  023326                    DT11
1640  002154  023452                    DF11
1641
1642                            ;ERROR 12
1643
1644  002156  022243                    EM12            ;DISK ERROR WHILE VERIFYING POSITION
1645  002160  022732                    DH12
1646  002162  023344                    DT12
1647  002164  023456                    DF12
1648
1649                            ;ERROR 13
1650
1651  002166  022307                    EM13            ;DISK POSITIONED TO WRONG CYLINDER
1652  002170  023027                    DH13
1653  002172  023364                    DT13
1654  002174  023462                    DF13
1655
1656                            ;ERROR 14
1657
1658  002176  022351                    EM14            ;NO INTERRUPT FROM I/O AFTER 1 SECOND
1659  002200  022570                    DH2
1660  002202  023312                    DT2
1661  002204  023446                    DF2
1662
1663                            ;ERROR 15
1664
1665  002206  022416                    EM15            ;SEEK INCOMPLETE
```

```
1666  002210  022570                        DH2
1667  002212  023312                        DT2
1668  002214  023446                        DF2
1669
1670                              ;ERROR 16
1671
1672  002216  022436                        EM16          ;DRIVE NOT READY AFTER POSITIONING
1673  002220  022570                        DH2
1674  002222  023312                        DT2
1675  002224  023446                        DF2
1676
1677                              ;ERROR 17
1678
1679  002226  022500                        EM17          ;DRIVE NOT READY/OFFLINE DURING TIMING TEST
1680  002230  022570                        DH2
1681  002232  023312                        DT2
1682  002234  023446                        DF2
1683
1684
```

```
                                        .SBTTL  START OF PROGRAM

002236  012737  177777  001224  START3: MOV     #-1,@#BUSADR    ;GET BUSADR FLAG
002244  000402                  START1: BR      STRT1A
002246  005037  001224          START1: CLR     @#BUSADR        ;CLR BUSADR FLAG
002252  005037  001222          STRT1A: CLR     @#CNTRLC        ;NO CONTROL "C"
002256  000411                  START:  BR      START           ;
002260  012737  177777  001224  START4: MOV     #-1,@#BUSADR    ;SET BUSADR FLAG
002266  000402                  START2: BR      STRT2A
002270  005037  001224          START2: CLR     @#BUSADR        ;CLR BUSADR FLAG
002274  012737  177777  001222  STRT2A: MOV     #-1,@#CNTRLC    ;SET CONTROL "C" FLAG
002302  012737  177777  001264  START:  MOV     #-1,STR20X      ;SET START FLAG
002310  000005                  START:  RESET
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002312  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002316  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002320  022706  001140                  CMP     #SWR,R6 ;;DONE?
002324  001374                          BNE     .-6             ;;LOOP BACK IF NO
002326  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
002332  012737  012502  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002340  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002346  012737  007732  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002354  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002362  012737  013134  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002370  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002376  005037  001206                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002402  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002406  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002414  012737  002414  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002422  012737  002422  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002430  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002434  012737  002470  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002442  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002450  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002456  022777  177777  176454          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002464  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002466  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002470  012716  002476          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002474  000002                          RTI
002476  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002504  012737  000174  001142          MOV     #DISPREG,DISPLAY
002512  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002516  012700  001160                  MOV     #$REGAD,R0      ;FIRST ADDRESS
002522  005020                  1$:     CLR     (R0)+           ;CLEAR VARIABLE STORAGE
002524  022700  001212                  CMP     #$BELL,R0       ;DONE?
002530  001374                          BNE     1$              ;NO--BRANCH
002532  005227  177777                  INC     #-1             ;FIRST START ?
002536  001032                          BNE     2$              ;BR IF NOT
```

```
1741  002540  104401  002546                    TYPE    ,68$            ;;TYPE ASCIZ STRING
1742  002544  000427                            BR      67$             ;;GET OVER THE ASCIZ
1743                                    ;;68$:  .ASCIZ  <15><12>/MD-11-DZRPZ-8, RP11E DRIVE POSITIONING TEST/
1744  002624                            67$:
1745  002624  004737  011232                  2$:     JSR     PC,$TKINT       ;SETUP THE TTY KEYBOARD
1746                                    .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1747  002630  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1748  002634  001006                              BNE     69$             ;;BRANCH IF YES
1749  002636  023727  001140  000176              CMP     $WR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1750  002644  001005                              BNE     70$             ;;BRANCH IF NO
1751  002646  104406                            GTSWR                   ;;GET SOFT-SWR SETTINGS
1752  002650  000403                            BR      70$
1753  002652  112737  000001  001134  69$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1754  002660                            70$:
1755  002660  012737  000377  001234            MOV     #377,TSTNMS     ;SELECT TESTS 0-7
1756  002666  012700  001460                    MOV     #DFLT,R0        ;DEFAULT PARAMETERS POINTER
1757  002672  012701  001632                    MOV     #PRMD,R1        ;TABLE POINTER
1758  002676  010102                            MOV     R1,R2           ;STOP ADDRESS
1759  002700  012021                    3$:     MOV     (R0)+,(R1)+     ;MOVE DEFAULT PARAMETERS INTO
1760  002702  020002                            CMP     R0,R2           ;RUN TIME TABLES ** DONE?
1761  002704  103775                            BLO     3$              ;NO--BRANCH
1762  002706  000406                            BR      $RTINT          ;FINISH SETUP
1763  002710  000005                    START5: RESET                   ;'CONTROL C' RESET
1764  002712  004737  011232                    JSR PC,$TKINT
1765  002716  012737  177777  001222            MOV     #-1,CNTRLC      ;SET 'CONTROL C' SWITCH
1766  002724  012737  000340  000062  $RTINT: MOV     #PR7,TKVEC+2    ;SET TTY PRIORITY TO 7
1767  002732  005037  177776                    CLR     PS              ;SET PRIORITY TO ZERO
1768  002736  005037  001252                    CLR     DRVBAD          ;CLEAR OFFLINE/UNSAFE DRIVE BITS
1769  002742  005037  001206                    CLR     $TIMES          ;INITIALIZE NUMBER OF ITERATIONS
1770  002746  005037  001210                    CLR     $ESCAPE         ;CLEAR THE ESCAPE ON ERROR ADDRESS
1771  002752  005037  001244                    CLR     BYPASS          ;CLEAR THE BYPASS ADDRESS
1772  002756  012737  000001  001104            MOV     #1,$ICNT        ;PRESET ITERATION COUNT TO 1
1773  002764  112737  000001  001115            MOVB    #1,$ERMAX       ;ALLOW ONE ERROR PER TEST
1774  002772  012737  002772  001106            MOV     #.,$LPADR       ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1775  003000  012737  003000  001110            MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
1776  003006  004737  017226                    JSR     PC,GETADR       ;CHECK RP11 ADDRESS
1777  003012  004737  013612                    JSR     PC,RPINIT       ;FIND OUT WHICH DRIVES ARE ON SYSTEM
1778  003016  012737  000312  001236            MOV     #202.,MAXCYL    ;ASSUME RP02'S
1779  003024  005737  001240                    TST     DRVTYP          ;WHICH DRIVES ?
1780  003030  001403                            BEQ     1$              ;BR IF THEY REALLY WERE RP02'S
1781  003032  012737  000625  001236            MOV     #405.,MAXCYL    ;SET MAX CYLINDER FOR RP03'S
1782  003040  013737  001236  001436  1$:      MOV     MAXCYL,PRMLMT+2 ;UPDATE 'FC' LIMIT
1783  003046  013737  001235  001440            MOV     MAXCYL,PRMLMT+4 ;UPDATE 'LC' LIMIT
1784  003054  005737  001264                    TST     STR2OX          ;COMING FROM A 'CONTROL C' RESTART ?
1785  003060  001002                            BNE     9$              ;BR IF NOT
1786  003062  000137  003432                    JMP     START6          ;BYPASS 'LC' UPDATE AND DRIVE STATUS TYPEOUT
1787  003066                            9$:
1788  003066  013737  001236  001646            MOV     MAXCYL,LC1      ;SETUP MAX CYLINDER FOR TEST 1
1789  003074  013737  001236  001662            MOV     MAXCYL,LC2      ;SETUP MAX CYLINDER FOR TEST 2
1790  003102  013737  001236  001712            MOV     MAXCYL,LC4      ;SETUP MAX CYLINDER FOR TEST 4
1791  003110  013737  001236  001726            MOV     MAXCYL,LC5      ;SETUP MAX CYLINDER FOR TEST 5
1792  003116  013737  001236  001742            MOV     MAXCYL,LC6      ;SETUP MAX CYLINDER FOR TEST 6
1793  003124  013737  001236  001756            MOV     MAXCYL,LC7      ;SETUP MAX CYLINDER FOR TEST 7
1794  003132  013737  001236  001776            MOV     MAXCYL,LC11     ;SETUP MAX CYLINDER FOR TEST 11
1795  003140  012737  000200  001676            MOV     #128.,LC3       ;ASSUME RP02
1796  003146  005737  001240                    TST     DRVTYP          ;RP02 OR RP03 ?
```

```
1797  003152  001403                          BEQ     2$                      ;BR IF RP02'S
1798  003154  012737  000400  001676          MOV     #256.,LC3               ;CHANGE VALUE FOR RP03
1799  003162  005227  177777          2$:     INC     #-1                     ;FIRST START ?
1800  003166  001404                          BEQ     10$                     ;BR IF IT IS
1801  003170  032777  000200  175742          BIT     #SW07,@SWR              ;INHIBIT THE STATUS TYPEOUT
1802  003176  001115                          BNE     START6                  ;BR IF YES
1803  003200                          10$:
1804  003200  104401  003206                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
1805  003204  000412                          BR      64$                     ;;GET OVER THE ASCIZ
1806                                  ;;65$:  .ASCIZ  <15><12><12>/DRIVE STATUS:/<15><12><12>
1807  003232                          64$:
1808  003232  005001                          CLR     R1                      ;CLEAR TABLE POINTER
1809  003234  012702  000010                  MOV     #8.,R2                  ;COUNTER
1810  003240                          3$:
1811  003240  010146                          MOV     R1,-(SP)                ;;SAVE R1 FOR TYPEOUT
1812                                                                           ;;TYPE DRIVE NUMBER
1813  003242  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
1814  003244     001                          .BYTE   1                       ;;TYPE 1 DIGIT(S)
1815  003245     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
1816  003246  105761  001304                  TSTB    DRVSTA(R1)              ;CHECK DRIVE'S STATUS
1817  003252  001450                          BEQ     7$                      ;DRIVE IS OFFLINE OR NON-EXISTENT
1818  003254  100412                          BMI     4$                      ;DRIVE IS UNSAFE
1819  003256  104401  003264                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
1820  003262  000406                          BR      66$                     ;;GET OVER THE ASCIZ
1821                                  ;;67$:  .ASCIZ  /     ONLINE/
1822  003300                          66$:
1823  003300  000411                          BR      5$                      ;
1824  003302                          4$:
1825  003302  104401  003310                  TYPE    ,69$                    ;;TYPE ASCIZ STRING
1826  003306  000406                          BR      68$                     ;;GET OVER THE ASCIZ
1827                                  ;;69$:  .ASCIZ  /     UNSAFE/
1828  003324                          68$:
1829  003324  136137  001314  001240  5$:     BITB    ATABIT(R1),DRVTYP       ;SEE WHICH DRIVE TYPE
1830  003332  001010                          BNE     6$                      ;BR IF RP03
1831  003334  104401  003342                  TYPE    ,71$                    ;;TYPE ASCIZ STRING
1832  003340  000404                          BR      70$                     ;;GET OVER THE ASCIZ
1833                                  ;;71$:  .ASCIZ  /  RP02/
1834  003352                          70$:
1835  003352  000422                          BR      8$                      ;
1836  003354                          6$:
1837  003354  104401  003362                  TYPE    ,73$                    ;;TYPE ASCIZ STRING
1838  003360  000404                          BR      72$                     ;;GET OVER THE ASCIZ
1839                                  ;;73$:  .ASCIZ  /  RP03/
1840  003372                          72$:
1841  003372  000412                          BR      8$                      ;
1842  003374                          7$:
1843  003374  104401  003402                  TYPE    ,75$                    ;;TYPE ASCIZ STRING
1844  003400  000407                          BR      74$                     ;;GET OVER THE ASCIZ
1845                                  ;;75$:  .ASCIZ  /     OFFLINE/
1846  003420                          74$:
1847  003420  104401  001217          8$:     TYPE    ,$CRLF                  ;CR-LF
1848  003424  005201                          INC     R1                      ;INCREMENT TABLE POINTER
1849  003426  005302                          DEC     R2                      ;DECREMENT THE COUNTER
1850  003430  001303                          BNE     3$                      ;CONTINUE
1851  003432  005037  001264          START6: CLR     STR20X                  ;CLEAR MANUAL START FLAG
1852  003436  005737  001222                  TST     CNTRLC                  ;CONTROL "C" START/RESTART?
```

```
 1853  003442  001403                          BEQ     1$                      ;NO--BRANCH
 1854  003444  004737  017560                  JSR     PC,@#GT.PRM             ;YES--GET PARAMETERS
 1855  003450  000416                          BR      5$
 1856  003452  005037  001232          1$:     CLR     DRVSEL                  ;NO DRIVES SELECTED
 1857  003456  005000                          CLR     R0                      ;DETERMINE THE DRIVES THAT
 1858  003460  012701  000001                  MOV     #1,R1                   ;ARE AVAILABLE FOR TESTING
 1859  003464  105760  001304          2$:     TSTB    DRVSTA(R0)
 1860  003470  003403                          BLE     3$
 1861  003472  156037  001314  001232          BISB    ATABIT(R0),@#DRVSEL
 1862  003500  005200                  3$:     INC     R0
 1863  003502  106301                          ASLB    R1
 1864  003504  001367                          BNE     2$
 1865  003506                          5$:
 1866  003506  104401  003514                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
 1867  003512  000415                          BR      64$                     ;;GET OVER THE ASCIZ
 1868                                  ;;65$:  .ASCIZ  <15><12>/DRIVE(S) TO BE TESTED /
 1869  003546                          64$:
 1870  003546  005037  007646                  CLR     @#SENDCT                ;DETERMINE PASSES TO MAKE AND
 1871  003552  005000                          CLR     R0                      ;THE DRIVES TO BE TESTED
 1872  003554  013701  001232                  MOV     @#DRVSEL,R1             ;ANY DRIVES SELECTED?
 1873  003560  001010                          BNE     9$                      ;YES--BRANCH
 1874  003562  104401  003570                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
 1875  003566  000403                          BR      66$                     ;;GET OVER THE ASCIZ
 1876                                  ;;67$:  .ASCIZ  /NONE/
 1877  003576                          66$:
 1878  003576  000137  007464          9$:     JMP     @#SEOP                  ;GO TO END OF PROGRAM
 1879  003602  005737  001222                  TST     CNTRLC                  ;CONTROL "C" START/RESTART ?
 1880  003606  001010                          BNE     6$                      ;BR IF NOT
 1881  003610  005737  000042                  TST     42                      ;UNDER MONITOR CONTROL ?
 1882  003614  001405                          BEQ     6$                      ;BR IF NOT
 1883  003616  122737  000007  000041          CMPB    #7,41                   ;LOADED BY RP02/RP03 ?
 1884  003624  001001                          BNE     6$                      ;BR IF NOT
 1885  003626  006201                          ASR     R1                      ;EXCLUDE DRIVE 0 FROM TESTING
 1886  003630  006201                  6$:     ASR     R1                      ;REPORT THE DRIVES TO BE TESTED
 1887  003632  103013                          BCC     7$
 1888  003634  005237  007646                  INC     @#SENDCT                ;GIVE THIS DRIVE A PASS
 1889  003640  010046                          MOV     R0,-(SP)                ;;SAVE R0 FOR TYPEOUT
 1890  003642  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
 1891  003644     001                          .BYTE   1                       ;;TYPE 1 DIGIT(S)
 1892  003645     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
 1893  003646  005701                          TST     R1                      ;MORE DRIVES?
 1894  003650  001406                          BEQ     8$                      ;NO--BRANCH
 1895  003652  104401  003660                  TYPE    ,69$                    ;;TYPE ASCIZ STRING
 1896  003656  000401                          BR      68$                     ;;GET OVER THE ASCIZ
 1897                                  ;;69$:  .ASCIZ  ","
 1898  003662                          68$:
 1899  003662  005200                  7$:     INC     R0                      ;INCREMENT DRIVE NUMBER
 1900  003664  000761                          BR      6$
 1901  003666  013737  007646  007640  8$:     MOV     @#SENDCT,@#SEOPCT
 1902  003674  005037  001246          RSTRT1: CLR     CHKDRV                  ;INIT. THE CHECK DRIVE KEY
 1903  003700  012737  000001  001250          MOV     #1,DRVMSK               ;START TO CHECK DESIRED DRIVES
 1904  003706  033737  001250  001232  RSTRT2: BIT     DRVMSK,DRVSEL           ;IS THIS DRIVE SELECTED?
 1905  003714  001011          RSTRT3: BNE     DRVOK                   ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
 1906  003716  005237  001246  RESTART:INC     CHKDRV                  ;MOVE  TO NEXT DRIVE NUMBER
 1907  003722  106337  001250          ASLB    DRVMSK                  ;POSITION THE MASK
 1908  003726  103367                  BCC     RSTRT2                  ;BR IF MORE DRIVES
```

```
 1909  003730  043737  001252  001232          BIC     DRVBAD,DRVSEL   ;CLEAR SELECTION BITS FOR ANY OFFLINE/UNSAFE
 1910                                                                  ;DRIVES
 1911  003736  000756                          BR      RSTRT1          ;CONTINUE WITH CYCLE
 1912  003740  013702  001246          DRVOK:  MOV     @#CHKDRV,R2     ;PICKUP THE DRIVE NUMBER
 1913  003744  110237  001365                  MOVB    R2,DPB+1        ;LOAD DRIVE NUMBER
 1914  003750  104401  003756                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1915  003754  000411                          BR      64$             ;;GET OVER THE ASCIZ
 1916                                  ;;65$:  .ASCIZ  <15><12><12>/TESTING DRIVE /
 1917  004000                          64$:
 1918  004000  010246                          MOV     R2,-(SP)        ;;SAVE R2 FOR TYPEOUT
 1919  004002  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 1920  004004     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
 1921  004005     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 1922  004006  104401  001217                  TYPE    ,$CRLF
 1923  004012  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
 1924  004016  016037  000012  001254          MOV     RPCA(R0),CYL.CR ;INITIALIZE CURRENT STORAGE
```

```
1925
1926                                    .SBTTL  #### TESTS ####
1927

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

                ;*IN THE DESCRIPTIONS OF THE BELOW TESTS THE VARIABLES USED
                ;*AND THEIR DEFAULT VALUES (UNLESS SPECIFIED OTHERWISE) ARE:
                ;*
                ;*MNEMONIC       VALUE           VARIABLE
                ;*--------       -----           --------
                ;*
                ;*R              1               ITERATIONS (REPEATS)
                ;*FC             0               FIRST CYLINDER ADDRESS
                ;*LC             410             LAST CYLINDER ADDRESS
                ;*IC             1               INCREMENT VALUE
                ;*NC OR NC1      FC+IC           NEW OR MODIFIED CYLINDER
                ;*                               ADDRESS
                ;*NC2            LC-IC           NEW OR MODIFIED CYLINDER
                ;*                               ADDRESS
                ;*
                ;*TK             0               TRACK ADDRESS

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

1954
1955
1956                                    ;;*****************************************************************
1957                                    ;*TEST 0         HOME SEEK TEST
1958
1959                                    ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A HOME SEEK
1960                                    ;*      COMMAND CYCLE.  AT THE COMPLETION OF THE COMMAND, THE
1961                                    ;*      STATUS INDICATORS ARE CHECKED TO VERIFY THAT NO ERRORS
1962                                    ;*      HAVE OCCURED.
1963                                    ;*      THIS TEST IS REPEATED 100 TIMES.
1964
1965                                    ;;*****************************************************************
1966  004024                            TSTO:
1967  004024  033737  001324  001234            BIT     @#BITS+<0*2>,TSTNMS ;DO THIS TEST?
1968  004032  001002                            BNE     64$             ;YES--BRANCH
1969  004034  000137  004146                    JMP     TST1            ;NO--GO TO THE NEXT TEST
1970  004040  013737  001634  001206  64$:     MOV     @#RPT0,$TIMES   ;GET THE ITERATION COUNT
1971  004046  012737  004134  001106            MOV     #TEST0,@#$LPADR
1972  004054  012737  004024  001110            MOV     #TST0,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
1973  004062  012737  000000  001102            MOV     #0,@#$TSTNM     ;SET UP TEST NUMBER AND
1974                                                                    ;CLEAR THE ERROR FLAG (SERFLG)
1975  004070  013777  001102  175044            MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
1976  004076  012737  004144  001244            MOV     #EXIT0,BYPASS   ;SETUP BYPASS ADDRESS
1977  004104  013700  001266                    MOV     RPADR,R0        ;RP11 ADDRESS
1978  004110  112737  000015  001364            MOVB    #HOMSEK,DPB     ;LOAD HOME SEEK OPCODE
1979  004116  005037  001366                    CLR     DPB+2           ;CYL ADDR ZERO
1980  004122  005037  001370                    CLR     DPB+4           ;CLEAR TRACK/SECTOR STORAGE
```

```
 1981  004126  113737  001636  001371          MOVB    TKO,DPB+5       ;LOAD USER SPECIFIED TRACK
 1982  004134  012706  001100          TEST0:  MOV     #STACK,SP       ;SET UP STACK POINTER
 1983  004140  004737  014006                  JSR     PC,SEEKS        ;DO THE SPECIFIED SEEK
 1984  004144  000004                  EXIT0:  SCOPE                   ;LOOP
 1985
 1986                                          ;;*****************************************************************
 1987                                          ;*TEST 1         SEEK/SEEK TEST
 1988
 1989                                          ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
 1990                                          ;*      CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
 1991                                          ;*      "FC", "FT", "FS".  AT THE COMPLETION OF EACH SEEK, THE PROPER
 1992                                          ;*      INDICATORS ARE EXAMINED TO INSURE PROPER OPERATION.
 1993
 1994                                          ;;*****************************************************************
 1995  004146                          TST1:
 1996  004146  033737  001326  001234          BIT     @#BITS+(1*2),TSTNMS ;DO THIS TEST?
 1997  004154  001002                          BNE     64$             ;YES--BRANCH
 1998  004156  000137  004304                  JMP     TST2            ;NO--GO TO THE NEXT TEST
 1999  004162  013737  001642  001206  64$:    MOV     @#RPT1,$TIMES   ;GET THE ITERATION COUNT
 2000  004170  012737  004252  001106          MOV     #TEST1,@#$LPADR
 2001  004176  012737  004146  001110          MOV     #TST1,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
 2002  004204  012737  000001  001102          MOV     #1,@#$TSTNM     ;SET UP TEST NUMBER AND
 2003                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
 2004  004212  013777  001102  174722          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2005  004220  012737  004302  001244          MOV     #EXIT1,BYPASS   ;SETUP BYPASS ADDRESS
 2006  004226  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
 2007  004232  112737  000011  001364          MOVB    #SEEK,DPB       ;LOAD SEEK OPCODE
 2008  004240  005037  001370                  CLR     DPB+4           ;CLEAR SECTOR/TRACK ADDRESS
 2009  004244  113737  001652  001371          MOVB    TK1,DPB+5       ;LOAD USER SUPPLIED TRACK ADDRESS
 2010  004252  012706  001100          TEST1:  MOV     #STACK,SP       ;SET THE STACK POINTER
 2011  004256  013737  001644  001366          MOV     FC1,DPB+2       ;STARTING CYLINDER
 2012  004264  004737  014006                  JSR     PC,SEEKS        ;SEEK TO FC
 2013  004270  013737  001646  001366          MOV     LC1,DPB+2       ;ENDING CYLINDER
 2014  004276  004737  014006                  JSR     PC,SEEKS        ;SEEK TO LC
 2015  004302  000004                  EXIT1:  SCOPE                   ;LOOP
 2016
 2017                                          ;;*****************************************************************
 2018                                          ;*TEST 2         INCREMENTAL SEEK TEST
 2019
 2020                                          ;*      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
 2021                                          ;*      CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC".
 2022                                          ;*      WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS
 2023                                          ;*      "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING
 2024                                          ;*      AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC"
 2025                                          ;*      UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH
 2026                                          ;*      SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
 2027                                          ;*      INSURE PROPER OPERATION.
 2028
 2029                                          ;;*****************************************************************
 2030  004304                          TST2:
 2031  004304  033737  001330  001234          BIT     @#BITS+(2*2),TSTNMS ;DO THIS TEST?
 2032  004312  001002                          BNE     64$             ;YES--BRANCH
 2033  004314  000137  004476                  JMP     TST3            ;NO--GO TO THE NEXT TEST
 2034  004320  013737  001656  001206  64$:    MOV     @#RPT2,$TIMES   ;GET THE ITERATION COUNT
 2035  004326  012737  004410  001106          MOV     #TEST2,@#$LPADR
 2036  004334  012737  004304  001110          MOV     #TST2,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
```

```
2037  004342  012737  000002  001102          MOV     #2,@#$TSTNM     ;SET UP TEST NUMBER AND
2038                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
2039  004350  013777  001102  174564          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2040  004356  012737  004474  001244          MOV     #EXIT2,BYPASS  ;SETUP BYPASS ADDRESS
2041  004364  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
2042  004370  112737  000011  001364          MOVB    #SEEK,DPB       ;LOAD SEEK OPCODE
2043  004376  005037  001370                  CLR     DPB+4           ;CLEAR SECTOR/TRACK ADDRESS
2044  004402  113737  001666  001371          MOVB    TK2,DPB+5       ;LOAD USER SUPPLIED TRACK ADDRESS
2045  004410  012706  001100          TEST2:  MOV     #STACK,SP       ;SET UP THE STACK POINTER
2046  004414  013737  001660  001366          MOV     FC2,DPB+2       ;STARTING CYLINDER
2047  004422  004737  014006          INCSK:  JSR     PC,SEEKS        ;SEEK TO FC
2048  004426  063737  001664  001366          ADD     IC2,DPB+2        ;MOVE TO NEXT CYLINDER
2049  004434  023737  001662  001366          CMP     LC2,DPB+2        ;OUT OF CYLINDERS?
2050  004442  002367                          BGE     INCSK           ;NO--BRANCH
2051  004444  013737  001662  001366          MOV     LC2,DPB+2       ;LOAD LC
2052  004452  004737  014006          DECSK:  JSR     PC,SEEKS        ;SEEK TO LC
2053  004456  163737  001664  001366          SUB     IC2,DPB+2
2054  004464  023737  001660  001366          CMP     FC2,DPB+2
2055  004472  003767                          BLE     DECSK
2056  004474  000004                  EXIT2:  SCOPE                   ;LOOP
2057
2058                                  ;;**************************************************************
2059                                  ;*TEST 3        STEPPING SEEK TEST
2060
2061                                  ;*      THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4,
2062                                  ;*      8, 16, 32, 64, AND 128 (AND 256 IF AN RP03).  AT THE
2063                                  ;*      COMPLETION OF EACH SEEK COMMAND, THE PROPER INDICATORS ARE
2064                                  ;*      EXAMINED TO VERIFY PROPER OPERATION.
2065
2066                                  ;;**************************************************************
2067  004476                          TST3:
2068  004476  033737  001332  001234          BIT     @#BITS+(3*2),TSTNMS ;DO THIS TEST?
2069  004504  001002                          BNE     64$             ;YES--BRANCH
2070  004506  000137  004646                  JMP     TST4            ;NO--GO TO THE NEXT TEST
2071  004512  013737  001672  001206  64$:    MOV     @#RPT3,$TIMES   ;GET THE ITERATION COUNT
2072  004520  012737  004602  001106          MOV     #TEST3,@#$LPADR
2073  004526  012737  004476  001110          MOV     #TST3,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
2074  004534  012737  000003  001102          MOV     #3,@#$TSTNM     ;SET UP TEST NUMBER AND
2075                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
2076  004542  013777  001102  174372          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2077  004550  012737  004644  001244          MOV     #EXIT3,BYPASS  ;SETUP BYPASS ADDRESS
2078  004556  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
2079  004562  112737  000011  001364          MOVB    #SEEK,DPB       ;LOAD SEEK OPCODE
2080  004570  005037  001370                  CLR     DPB+4           ;CLEAR SECTOR/TRACK ADDRESS
2081  004574  113737  001702  001371          MOVB    TK3,DPB+5       ;LOAD USER SUPPLIED TRACK ADDRESS
2082  004602  012706  001100          TEST3:  MOV     #STACK,SP       ;SET UP THE STACK
2083  004606  013737  001674  001366          MOV     FC3,DPB+2        ;FC
2084  004614  004737  014006                  JSR     PC,SEEKS        ;SEEK TO FC
2085  004620  013701  001700                  MOV     IC3,R1          ;CYLINDER 1
2086  004624  010137  001366          1$:     MOV     R1,DPB+2        ;DESIRED CYLINDER
2087  004630  004737  014006                  JSR     PC,SEEKS        ;SEEK TO NC
2088  004634  006301                          ASL     R1              ;MOVE TO NEXT CYLINDER
2089  004636  020137  001676                  CMP     R1,LC3          ;DONE?
2090  004642  003770                          BLE     1$              ;NO--LOOP
2091  004644  000004                  EXIT3:  SCOPE
2092
```

```
2093                                    ;;*****************************************************************
2094                                    ;*TEST 4        OSCILLATING SEEK TEST
2095                                    ;
2096                                    ;*      THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
2097                                    ;*      TO "FC".  "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
2098                                    ;*      "LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC".  AT THE
2099                                    ;*      COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
2100                                    ;*      EXAMINED TO VERIFY PROPER OPERATION.
2101                                    ;
2102                                    ;;*****************************************************************
2103  004646                            TST4:
2104  004646  033737  001334  001234            BIT     @#BITS+(4*2),TSTNMS ;DO THIS TEST?
2105  004654  001002                            BNE     64$                 ;YES--BRANCH
2106  004656  000137  005210                    JMP     TST5                ;NO--GO TO THE NEXT TEST
2107  004662  013737  001706  001206  64$:     MOV     @#RPT4,$TIMES       ;GET THE ITERATION COUNT
2108  004670  012737  004766  001106            MOV     #TEST4,@#$LPADR
2109  004676  012737  004646  001110            MOV     #TST4,@#$LPERR      ;SETUP THE ERROR LOOP ADDRESS
2110  004704  012737  000004  001102            MOV     #4,@#$TSTNM         ;SET UP TEST NUMBER AND
2111                                                                        ;CLEAR THE ERROR FLAG ($ERFLG)
2112  004712  013777  001102  174222            MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2113  004720  012737  005206  001244            MOV     #EXIT4,BYPASS       ;SETUP BYPASS ADDRESS
2114  004726  013700  001266                    MOV     RPADR,R0            ;RP11 ADDRESS
2115  004732  112737  000011  001364            MOVB    #SEEK,DPB           ;LOAD SEEK OPCODE
2116  004740  005037  001370                    CLR     DPB+4               ;CLEAR SECTOR/TRACK ADDRESS
2117  004744  113737  001716  001371            MOVB    TK4,DPB+5           ;LOAD USER SUPPLIED TRACK ADDRESS
2118  004752  005002                            CLR     R2                  ;CLEAR STALL SWITCH (NO STALL)
2119  004754  032777  000002  174156            BIT     #SW1,@SWR           ;STALL REQUIRED?
2120  004762  001401                            BEQ     TEST4               ;NO--BRANCH
2121  004764  005102                            COM     R2                  ;YES--SET SWITCH
2122  004766  012706  001100          TEST4:    MOV     #STACK,SP           ;SET UP THE STACK POINTER
2123  004772  013701  001710                    MOV     FC4,R1              ;SET NC TO FC
2124  004776  005037  001274                    CLR     STALLO              ;START AT ZERO IF STALLS REQUIRED
2125  005002  010137  001366          1$:       MOV     R1,DPB+2            ;NC
2126  005006  004737  014006                    JSR     PC,SEEKS            ;SEEK TO NC
2127  005012  005702                            TST     R2                  ;STALL?
2128  005014  001403                            BEQ     2$                  ;NO--BRANCH
2129  005016  004037  015546                    JSR     R0,STALL            ;YES-GO TO STALL ROUTINE
2130  005022  001274                            .WORD   STALLO              ;TIME POINTER
2131  005024  013737  001710  001366  2$:       MOV     FC4,DPB+2           ;FC
2132  005032  004737  014006                    JSR     PC,SEEKS            ;SEEK TO FC
2133  005036  005702                            TST     R2                  ;STALL?
2134  005040  001413                            BEQ     3$                  ;NO--BRANCH
2135  005042  004037  015546                    JSR     R0,STALL            ;YES--GO TO STALL ROUTINE
2136  005046  001274                            .WORD   STALLO              ;TIME POINTER
2137  005050  005237  001274                    INC     STALLO              ;UPDATE THE TIME
2138  005054  023737  001302  001274            CMP     MXSTAL,STALLO       ;TIME TO BIG?
2139  005062  003347                            BGT     1$                  ;NO--BRANCH
2140  005064  005037  001274                    CLR     STALLO              ;YES--START OVER AT ZERO
2141  005070  063701  001714          3$:       ADD     IC4,R1              ;MOVE TO NEXT CYLINDER
2142  005074  020137  001712                    CMP     R1,LC4              ;LAST CYLINDER COMPLETED?
2143  005100  003740                            BLE     1$                  ;NO--BRANCH
2144  005102  013701  001712                    MOV     LC4,R1              ;SET NC TO LC
2145  005106  010137  001366          4$:       MOV     R1,DPB+2            ;NC
2146  005112  004737  014006                    JSR     PC,SEEKS            ;SEEK TO NC
2147  005116  005702                            TST     R2                  ;STALL?
2148  005120  001403                            BEQ     5$                  ;NO--BRANCH
```

```
2149  005122  004037  015546                  JSR     R0,STALL        ;YES--GO TO STALL ROUTINE
2150  005126  001274                          .WORD   STALLO          ;TIME POINTER
2151  005130  013737  001712  001366  5$:     MOV     LC4,DPB+2       ;LC
2152  005136  004737  014006                  JSR     PC,SEEKS        ;SEEK TO LC
2153  005142  005702                          TST     R2              ;STALL?
2154  005144  001413                          BEQ     6$              ;NO--BRANCH
2155  005146  004037  015546                  JSR     R0,STALL        ;YES--GO TO STALL ROUTINE
2156  005152  001274                          .WORD   STALLO          ;TIME POINTER
2157  005154  005237  001274                  INC     STALLO          ;UPDATE STALL TIME
2158  005160  023737  001302  001274          CMP     MXSTAL,STALLO   ;TIME TOO BIG?
2159  005166  003347                          BGT     4$              ;NO--BRANCH
2160  005170  005037  001274                  CLR     STALLO          ;YES--SET STALL TIME BACK TO ZERO
2161  005174  163701  001714          6$:     SUB     IC4,R1          ;NEXT CYLINDER
2162  005200  020137  001710                  CMP     R1,FC4          ;DONE?
2163  005204  002373                          BGE     6$              ;NO--BRANCH
2164  005206  000004                  EXIT4:  SCOPE                   ;LOOP
2165
2166                                  ;;**************************************************************
2167                                  ;*TEST 5        CONVERGING/DIVERGING SEEK TEST
2168
2169                                  ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
2170                                  ;*      SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
2171                                  ;*      BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
2172                                  ;*      GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
2173                                  ;*      LESS THAN THE INITIAL VALUE OF "NC1".  AT THE COMPLETION OF
2174                                  ;*      EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
2175                                  ;*      VERIFY PROPER OPERATION.  "NC1" AND "NC2" DEFAULT TO
2176                                  ;*      "FC" AND "LC" RESPECTIVELY.
2177
2178                                  ;;**************************************************************
2179  005210                          TST5:
2180  005210  033737  001336  001234          BIT     @#BITS+(5*2),TSTNMS ;DO THIS TEST?
2181  005216  001002                          BNE     64$             ;YES--BRANCH
2182  005220  000137  005376                  JMP     TST6            ;NO--GO TO THE NEXT TEST
2183  005224  013737  001722  001206  64$:     MOV     @#RPT5,$TIMES   ;GET THE ITERATION COUNT
2184  005232  012737  005314  001106          MOV     #TEST5,@#$LPADR
2185  005240  012737  005210  001110          MOV     #TST5,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
2186  005246  012737  000005  001102          MOV     #5,@#$TSTNM     ;SET UP TEST NUMBER AND
2187                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
2188  005254  013777  001102  173660          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2189  005262  012737  005374  001244          MOV     #EXIT5,BYPASS   ;SETUP BYPASS ADDRESS
2190  005270  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
2191  005274  112737  000011  001364          MOVB    #SEEK,DPB       ;LOAD SEEK OPCODE
2192  005302  005037  001370                  CLR     DPB+4           ;CLEAR SECTOR/TRACK ADDRESS
2193  005306  113737  001732  001371          MOVB    TK5,DPB+5       ;LOAD USER SUPPLIED TRACK ADDRESS
2194  005314  012706  001100          TEST5:  MOV     #STACK,SP
2195  005320  013701  001724                  MOV     FC5,R1          ;START NC1 AT FC
2196  005324  013702  001726                  MOV     LC5,R2          ;START NC2 AT LC
2197  005330  010137  001366          1$:     MOV     R1,DPB+2        ;NC1
2198  005334  004737  014006                  JSR     PC,SEEKS        ;SEEK TO NC1
2199  005340  010237  001366                  MOV     R2,DPB+2        ;NC2
2200  005344  004737  014006                  JSR     PC,SEEKS        ;SEEK TO NC1
2201  005350  063701  001730                  ADD     IC5,R1          ;NEXT NC1
2202  005354  163702  001730                  SUB     IC5,R2          ;NEXT NC2
2203  005360  020137  001726                  CMP     R1,LC5          ;DONE?
2204  005364  003003                          BGT     EXIT5           ;YES--BRANCH
```

```
  2205  005366  020237  001724                CMP     R2,FC5          ;?
  2206  005372  002356                        BGE     1$              ;NO--BRANCH
  2207  005374  000004                EXIT5:  SCOPE                   ;LOOP
  2208
  2209                                ;;*****************************************************************
  2210                                ;*TEST 6         SERVO ADDRESSING LOGIC NOISE GENERATOR
  2211
  2212                                ;*      IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
  2213                                ;*      NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5.  NOW "NC" IS UPDATED
  2214                                ;*      BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNITL "LC" IS
  2215                                ;*      EXCEEDED BY ANY OF THE ABOVE VALUES.  THE INITIAL VALUE OF "NC"
  2216                                ;*      IS "FC".  AT THE COMPLETION OF EACH SEEK COMMAND THE
  2217                                ;*      PROPER INDICATORS ARE EXAMINED TO INSURE PROPER OPERATION.
  2218
  2219                                ;;*****************************************************************
  2220  005376                        TST6:
  2221  005376  033737  001340  001234        BIT     @#BITS+(6*2),TSTNMS ;DO THIS TEST?
  2222  005404  001002                        BNE     64$             ;YES--BRANCH
  2223  005406  000137  005630                JMP     TST7            ;NO--GO TO THE NEXT TEST
  2224  005412  013737  001736  001206  64$:  MOV     @#RPT6,$TIMES   ;GET THE ITERATION COUNT
  2225  005420  012737  005502  001106        MOV     #TEST6,@#SLPADR
  2226  005426  012737  005376  001110        MOV     #TST6,@#SLPERR  ;SETUP THE ERROR LOOP ADDRESS
  2227  005434  012737  000006  001102        MOV     #6,@#STSTNM     ;SET UP TEST NUMBER AND
  2228                                                                ;CLEAR THE ERROR FLAG (SERFLG)
  2229  005442  013777  001102  173472        MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
  2230  005450  012737  005626  001244        MOV     #EXIT6,BYPASS   ;SETUP BYPASS ADDRESS
  2231  005456  013700  001266                MOV     RPADR,R0        ;RP11 ADDRESS
  2232  005462  112737  000011  001364        MOVB    #SEEK,DPB       ;LOAD SEEK OPCODE
  2233  005470  005037  001370                CLR     DPB+4           ;CLEAR SECTOR/TRACK ADDRESS
  2234  005474  113737  001746  001371        MOVB    TK6,DPB+5       ;LOAD USER SUPPLIED TRACK ADDRESS
  2235  005502  012706  001100        TEST6:  MOV     #STACK,SP       ;SETUP STACK
  2236  005506  013701  001740                MOV     FC6,R1          ;PICKUP "FC"
  2237  005512  013702  001742                MOV     LC6,R2          ;FORM LAST CYLINDER THAT
  2238  005516  162702  000005                SUB     #5,R2           ;IS AVAILABLE FOR TESTING
  2239  005522  020102                1$:     CMP     R1,R2           ;LAST CYLINDER
  2240  005524  003040                        BGT     EXIT6           ;YES--BRANCH
  2241  005526  010137  001366                MOV     R1,DPB+2        ;NC
  2242  005532  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC
  2243  005536  062737  000004  001366        ADD     #4,DPB+2        ;NC+4
  2244  005544  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC+4
  2245  005550  162737  000003  001366        SUB     #3,DPB+2        ;NC+1
  2246  005556  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC+1
  2247  005562  062737  000002  001366        ADD     #2,DPB+2        ;NC+3
  2248  005570  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC+3
  2249  005574  162737  000001  001366        SUB     #1,DPB+2        ;NC+2
  2250  005602  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC+2
  2251  005606  062737  000003  001366        ADD     #3,DPB+2        ;NC+5
  2252  005614  004737  014006                JSR     PC,SEEK5        ;SEEK TO NC+5
  2253  005620  063701  001744                ADD     IC6,R1
  2254  005624  000736                        BR      1$
  2255  005626  000004                EXIT6:  SCOPE                   ;LOOP
  2256
  2257                                ;;*****************************************************************
  2258                                ;*TEST 7         CYLINDER ADDRESSING TEST
  2259
  2260                                ;*      THIS TEST WILL CAUSE THE DRIVE TO SEEK FROM EACH CYLINDER TO EVERY
```

```
2261                                    ;*      OTHER CYLINDER BETWEEN CYLINDERS "FC" AND "LC".  SEEK CYCLES ARE
2262                                    ;*      COMMANDED FROM CYLINDER "NC"TO CYLINDER "NC1" AND BACK TO CYLINER
2263                                    ;*      "NC".  "NC" AND "NC1" START AT "FC".  "NC1" IS INCREMENTED BY "IC"
2264                                    ;*      UNTIL "NC1" IS EQUAL TO "LC"; WHEN "NC1" IS EQUAL TO "LC", IT IS
2265                                    ;*      RESET TO "FC" AND "NC" IS INCREMENTED BY "IC".  THE CYCLE IS COMPLETE
2266                                    ;*      WHEN "NC" AND "NC1" BOTH EQUAL "LC".  AT THE COMPLETION OF EACH
2267                                    ;*      SEEK COMMAND, THE INDICATORS ARE EXAMINED TO VERIFY PROPER OPERATION.
2268
2269                                    ;;*****************************************************************
2270  005630                            TST7:
2271  005630  033737  001342  001234           BIT     @#BITS+(7*2),TSTNMS ;DO THIS TEST?
2272  005636  001002                           BNE     64$             ;YES--BRANCH
2273  005640  000137  006022                   JMP     TST10           ;NO--GO TO THE NEXT TEST
2274  005644  013737  001752  001206  64$:     MOV     @#RPT7,$TIMES   ;GET THE ITERATION COUNT
2275  005652  012737  005734  001106           MOV     #TEST7,@#$LPADR
2276  005660  012737  005630  001110           MOV     #TST7,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
2277  005666  012737  000007  001102           MOV     #7,@#$TSTNM     ;SET UP TEST NUMBER AND
2278                                            ;CLEAR THE ERROR FLAG ($ERFLG)
2279  005674  013777  001102  173240           MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2280  005702  012737  006020  001244           MOV     #EXIT7,BYPASS   ;SETUP BYPASS ADDRESS
2281  005710  013700  001266                   MOV     RPADR,R0        ;RP11 ADDRESS
2282  005714  112737  000011  001364           MOVB    #SEEK,DPB       ;SETUP FOR SEEKING
2283  005722  005037  001370                   CLR     DPB+4           ;CLEAR TRACK/SECTOR STORAGE
2284  005726  113737  001762  001371           MOVB    TK7,DPB+5       ;LOAD USER SPECIFIED TRACK ADDRESS
2285  005734  012706  001100           TEST7:  MOV     #STACK,SP       ;LOAD STACK POINTER
2286  005740  013701  001754                   MOV     FC7,R1          ;START NC AT FC
2287  005744  013702  001754                   MOV     FC7,R2          ;START NC1 AT FC
2288  005750  010137  001366           1$:     MOV     R1,DPB+2        ;NC
2289  005754  004737  014006                   JSR     PC,SEEKS        ;SEEK TO NC
2290  005760  010237  001366                   MOV     R2,DPB+2        ;NC1
2291  005764  004737  014006                   JSR     PC,SEEKS        ;SEEK TO NC1
2292  005770  063702  001760                   ADD     IC7,R2          ;INCREMENT NC1
2293  005774  023702  001756                   CMP     LC7,R2          ;EXCEEDED LC ?
2294  006000  002363                           BGE     1$              ;BR IF NOT
2295  006002  013702  001754                   MOV     FC7,R2          ;RESET NC1
2296  006006  063701  001760                   ADD     IC7,R1          ;INCREMENT NC
2297  006012  023701  001756                   CMP     LC7,R1          ;EXCEEDED LC ?
2298  006016  002354                           BGE     1$              ;BR IF NOT
2299  006020  000004                   EXIT7:  SCOPE                   ;LOOP ?
2300
2301                                    ;;*****************************************************************
2302                                    ;*TEST 10        ONE CYLINDER SEEK TIMER TEST
2303
2304                                    ;*      THIS TEST PERFORMS ONE CYLINDER SEEKS FROM CYLINDER 0 TO THE
2305                                    ;*      MAXIMUM CYLINDER FOR THE DRIVE (E.G. 202 FOR RP02'S AND 405 FOR
2306                                    ;*      RP03'S).  THE SEEK TIMES ARE MEASURED RELATIVE TO THE SECTOR
2307                                    ;*      PULSES FROM THE DRIVE.  THE TEST DISPLAYS THE FORWARD AND REVERSE
2308                                    ;*      TIMES.
2309
2310                                    ;;*****************************************************************
2311  006022                            TST10:
2312  006022  033737  001344  001234           BIT     @#BITS+(10*2),TSTNMS ;DO THIS TEST?
2313  006030  001002                           BNE     64$             ;YES--BRANCH
2314  006032  000137  006276                   JMP     TST11           ;NO--GO TO THE NEXT TEST
2315  006036  013737  001766  001206  64$:     MOV     @#RPT10,$TIMES  ;GET THE ITERATION COUNT
2316  006044  012737  006120  001106           MOV     #TEST10,@#$LPADR
```

D05


```
 2317  005052  012737  006022  001110          MOV     #TST10,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
 2318  006060  012737  000010  001102          MOV     #10,@#$TSTNM     ;SET UP TEST NUMBER AND
 2319                                                                   ;CLEAR THE ERROR FLAG ($ERFLG)
 2320  006066  013777  001102  173046          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2321  006074  012737  006274  001244          MOV     #EXIT10,BYPASS   ;SETUP BYPASS ADDRESS
 2322  006102  013700  001256                  MOV     RPADR,R0         ;RP11 ADDRESS
 2323  006106  112737  000011  001364          MOVB    #SEEK,DPB        ;OPCODE FOR THE TEST
 2324  006114  005037  001370                  CLR     DPB+4            ;TRACK ADDRESS
```

E05


```
 2325  006120  012706  001100          TEST10: MOV     #STACK,SP       ;RESTORE THE STACK POINTER
```

```
2326  006124  004737  015264                  JSR     PC,RESTOR       ;DO A HOME SEEK
2327  006130  012704  024010                  MOV     #FRWSTR,R4      ;FORWARD TIME STORAGE POINTER
2328  006134  012737  000001  001366          MOV     #1,DPB+2        ;SEEK TO THIS CYLINDER
2329  006142  004737  016476          1$:     JSR     PC,TIMSEK       ;START THE SEEK, AND TIME IT
2330  006146  010324                          MOV     R3,(R4)+        ;STORE THE TIME
2331  006150  005237  001366                  INC     DPB+2           ;INCREMENT THE CYLINDER ADDRESS
2332  006154  023737  001366  001236          CMP     DPB+2,MAXCYL    ;AT THE MAXIMUM ?
2333  006162  003767                          BLE     1$              ;BR IF NOT
2334  006164  013737  001236  001366          MOV     MAXCYL,DPB+2    ;SETUP TO SEEK BACK TO MINIMUM
2335  006172  005337  001366                  DEC     DPB+2           ;SEEK TO CYLINDER
2336  006176  012704  025464                  MOV     #RVSTR,R4       ;REVERSE TIME STORAGE POINTER
2337  006202  004737  016476          2$:     JSR     PC,TIMSEK       ;START THE SEEK, AND TIME IT
2338  006206  010324                          MOV     R3,(R4)+        ;STORE THE TIME
2339  006210  005337  001366                  DEC     DPB+2           ;DECREMENT THE CYLINDER ADDRESS
2340  006214  100372                          BPL     2$              ;BR IF NOT AT MINIMUM CYLINDER
2341  006216  104401  006224                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2342  006222  000416                          BR      64$             ;;GET OVER THE ASCIZ
2343                                  ;;65$:  .ASCIZ  <15><12><12>/ONE CYLINDER SEEK TIMES/
2344  006260                          64$:
2345  006260  013737  001236  006272          MOV     MAXCYL,3$       ;NUMBER OF SEEKS PERFORMED
2346  006266  004037  015630                  JSR     R0,SORT         ;SORT AND TYPE OUT THE TIMES
2347  006272  000000                  3$:     .WORD   0               ;NUMBER OF TIMES GOES HERE
2348  006274  000004                  EXIT10: SCOPE                   ;LOOP ?
2349
2350                                  ;;**************************************************************
2351                                  ;*TEST 11        SEEK-SEEK TIMER TEST
2352
2353                                  ;*      THIS TEST IS A GENERAL PURPOSE TIMING TEST WHICH PERFORMS 200
2354                                  ;*      ITERATIONS BETWEEN THE CYLINDER IN 'FC' AND THE CYLINDER IN 'LC'.
2355                                  ;*      BOTH FORWARD AND REVERSE SEEKS ARE TIMED AND DISPLAYED.  THE
2356                                  ;*      SEEK TIMES DISPLAYED ARE RELATIVE TO THE SECTOR COUNTER.
2357                                  ;*      'FC' DEFAULTS TO 0; 'LC' DEFAULTS TO 'MAXCYL' (202 IF RP02'S OR
2358                                  ;*      405 IF RP03'S).
2359
2360                                  ;;**************************************************************
2361  006276                          TST11:
2362  006276  033737  001346  001234          BIT     @#BITS+(11*2),TSTNMS ;DO THIS TEST?
2363  006304  001002                          BNE     64$             ;YES--BRANCH
2364  006306  000137  006610                  JMP     TST12           ;NO--GO TO THE NEXT TEST
2365  006312  013737  001772  001206  64$:     MOV     @#??T11,$TIMES  ;GET THE ITERATION COUNT
      006320  012737  006374  001106          MOV     #TEST11,@#$LPADR
      006326  012737  006276  001110          MOV     #TST11,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
      006334  012737  000011  001102          MOV     #11,@#$TSTNM    ;SET UP TEST NUMBER AND
                                              ;CLEAR THE ERROR FLAG ($ERFLG)
2369
2370  006342  013777  001102  172572          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2371  006350  012737  006606  001244          MOV     #EXIT11,BYPASS  ;SETUP BYPASS ADDRESS
2372  006356  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
2373  006362  012737  000011  001364          MOV     #SEEK,DPB       ;OPCODE FOR THE TEST
2374  006370  005037  001370                  CLR     DPB+4           ;CLEAR THE TRACK ADDRESS STORAGE
2375  006374  012706  001100          TEST11: MOV     #STACK,SP       ;RESTORE THE STACK POINTER
2376  006400  004737  015264                  JSR     PC,RESTOR       ;DO A HOME SEEK
2377  006404  012701  000311                  MOV     #201.,R1        ;ITERATION COUNT
2378  006410  012704  025464                  MOV     #RVSTR,R4       ;REVERSE TIME STORAGE POINTER
2379  006414  012705  024010                  MOV     #FRWSTR,R5      ;FORWARD TIME STORAGE POINTER
2380  006420  013737  001774  001366          MOV     FC11,DPB+2      ;POSITION THE DISK TO THE FIRST CYLINDER
2381  006426  004737  016476                  JSR     PC,TIMSEK       ;DO THE SEEK
```

```
 2382  006432  013737  001776  001366  1$:     MOV     LC11,DPB+2      ;TIME TO THIS CYLINDER
 2383  006440  004737  016476                  JSR     PC,TIMSEK       ;START THE SEEK AND TIME IT
 2384  006444  010325                          MOV     R3,(R5)+        ;STORE THE TIME
 2385  006446  013737  001774  001366          MOV     FC11,DPB+2      ;MINIMUM CYLINDER
 2386  006454  004737  016476                  JSR     PC,TIMSEK       ;START THE SEEK AND TIME IT
 2387  006460  010324                          MOV     R3,(R4)+        ;STORE THE TIME
 2388  006462  005301                          DEC     R1              ;DECREMENT THE ITERATION COUNTER
 2389  006464  001362                          BNE     1$              ;BR IF NOT FINISHED
 2390  006466  104401  006474                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 2391  006472  000421                          BR      64$             ;;GET OVER THE ASCIZ
 2392                                  ;;65$:   .ASCIZ  <15><12><12>/SEEK TIMES BETWEEN CYLINDERS /
 2393  006536                          64$:
 2394  006536  013746  001774                  MOV     FC11,-(SP)      ;PUT STARTING CYLINDER ADDRESS ON THE STACK
 2395  006542  004737  013220                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 2396  006546  004737  013450                  JSR     PC,$SUPRS       ;TYPE IT
 2397  006552  104401  006560                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 2399  006556  000402                          BR      66$             ;;GET OVER THE ASCIZ
 2399                                  ;;67$:   .ASCIZ  / & /
 2400  006564                          66$:
 2401  006564  013746  001776                  MOV     LC11,-(SP)      ;PUT ENDING CYLINDER ADDRESS ON THE STACK
 2402  006570  004737  013220                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 2403  006574  004737  013450                  JSR     PC,$SUPRS       ;TYPE IT
 2404  006600  004037  015630                  JSR     R0,$ORT         ;TYPE THE TIMES
 2405  006604  000310                          .WORD   200.
 2406  006606  000004                  EXIT11: SCOPE                   ;LOOP ?
 2407
 2408
 2409                                  ;;*************************************************
 2410                                  ;*TEST 12       SEEK TIME OF RPH TEST
 2411
 2412                                  ;*       THIS TEST PRODUCES TWO SEEK TIME GRAPHS.  THE FIRST GRAPH IS
 2413                                  ;*       TYPED IS SWITCH <04> IS 0.  THIS GRAPH IS A PLOT OF SEEK TIME
 2414                                  ;*       AGAINST CYLINDERS SEEKED TO FROM CYLINDER 0.  THE CYLINDERS
 2415                                  ;*       PLOTTED ARE: 0,1,2,3,4,6,8,10,12,14,16,18,20,25...100,105...MAXCYL.
 2416                                  ;*       THE OTHER GRAPH WHICH IS PRODUCED (SWITCH <04> SET) IS TIME TO
 2417                                  ;*       SEEK TO ALL CYLINDERS FROM CYLINDER 0.
 2418                                  ;*
 2419                                  ;*       IN BOTH SEEK GRAPHS, ONLY THE FORWARD SEEK TIMES ARE MEASURED.
 2420                                  ;*
 2421                                  ;*       THIS TEST USES THE SECTOR COUNTER ('SOT') IN THE RP11 AS THE
 2422                                  ;*       TIME BASE FOR THE MEASURMENTS, USE OF THE 'SOT' PROVIDES A
 2423                                  ;*       RESOLUTION OF 2.5 MS.
 2424
 2425                                  ;;*************************************************************
 2426  006610                          TST12:
 2427  006610  033737  001350  001234          BIT     @#BITS+<12*2>,TSTNMS ;DO THIS TEST?
 2428  006616  001002                          BNE     64$             ;YES--BRANCH
 2429  006620  000137  007464                  JMP     $EOP            ;NO--GO TO THE END OF THE PROGRAM
 2430  006624  013737  002002  001206  64$:    MOV     @#RPT12,$TIMES  ;GET THE ITERATION COUNT
 2431  006632  012737  006706  001106          MOV     #TEST12,@#$LPADR
 2432  006640  012737  006610  001110          MOV     #TST12,@#$LPERR  ;SETUP THE ERROR LOOP ADDRESS
 2433  006646  012737  000012  001102          MOV     #12,@#$TSTNM    ;SET UP TEST NUMBER AND
 2434                                                                  ;CLEAR THE ERROR FLAG (SERFLG)
 2435  006654  013777  001102  172260          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2436  006662  012737  007462  001244          MOV     #EXIT12,BYPASS  ;SETUP BYPASS ADDRESS
 2437  006670  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
```

```
2433  006674  112737  000011  001364          MOVB    #SEEK,DPB       ;OPCODE FOR THE TEST
2439  006702  005037  001370                  CLR     DPB+4           ;CLEAR THE TRACK ADDRESS STORAGE
2440  006706  012706  001100          TEST12: MOV     #STACK,SP       ;INITIALIZE THE STACK POINTER
2441  006712  004737  015264                  JSR     PC,RESTOR       ;DO A HOME SEEK
2442  006716  012704  024010                  MOV     #FRWSTR,R4      ;STORE 'SEEK TIME' HERE
2443  006722  005001                          CLR     R1              ;CLEAR 'FC' STORAGE
2444  006724  005002                          CLR     R2              ;CLEAR 'NC' STORAGE
2445  006726  010237  001366          1$:     MOV     R2,DPB+2        ;ADDRESS OF NEW CYLINDER 'NC'
2446  006732  004737  016476                  JSR     PC,TIMSEK       ;SEEK TO 'NC' AND TIME IT
2447  006736  010324                          MOV     R3,(R4)+        ;STORE MEASURED SEEK TIME
2448  006740  010137  001366                  MOV     R1,DPB+2        ;'FC' ADDRESS
2449  006744  004737  016476                  JSR     PC,TIMSEK       ;RETURN TO 'FC'
2450  006750  005202                          INC     R2              ;INCREMENT 'NC'
2451  006752  023702  001236                  CMP     MAXCYL,R2       ;IS 'NC' AT MAXIMUM ?
2452  006756  002363                          BGE     1$              ;BR IF NOT
2453
2454                                  ;PLOT A GRAPH USING MEASURED SEEK TIMES
2455
2456  006760                          PLOT:
2457  006760  104401  006766                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2458  006764  000412                          BR      64$             ;;GET OVER THE ASCIZ
2459                                  ;;65$:  .ASCIZ  <15><12><12>/SEEK TIME GRAPH/
2460  007012                          64$:
2461  007012  104401  007020                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2462  007016  000421                          BR      66$             ;;GET OVER THE ASCIZ
2463                                  ;;67$:  .ASCIZ  <15><12>/X AXIS - SEEK TIME - MILLI SECS/
2464  007062                          66$:
2465  007062  104401  007070                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2466  007066  000423                          BR      68$             ;;GET OVER THE ASCIZ
2467                                  ;;69$:  .ASCIZ  <15><12>/Y AXIS - CYLINDER SEEKED FROM 0/<15><12><12>
2468  007136                          68$:
2469  007136  104401  021443                  TYPE    ,GRPH1          ;TYPE THE GRAPH HEADING
2470  007142  032777  000020  171770          BIT     #SW4,@SWR       ;TYPE COMPLETE GRAPH?
2471  007150  001064                          BNE     CMPGRP          ;YES BRANCH; IF NOT, TYPE SMALL GRAPH
2472
2473                                  ;IN THE SMALL GRAPH, SEEK TIMES ARE PLOTTED ONLY FOR SELECTED CYLINDERS.
2474                                  ;THE CYLINDERS PLOTTED ARE: 0,1,2,3,4,6,8,10,12,14,16,18,20,25,30
2475                                  ;35,40...100,105...MAXCYL
2476
2477  007152  005000                  SMGRP:  CLR     R0
2478  007154  005037  024010                  CLR     FRWSTR          ;FORCE CYL 0   CYL 0 SEEK TO ZERO TIME
2479  007160  104401  001217          1$:     TYPE    ,$CRLF          ;CR-LF
2480  007164  010046                          MOV     R0,-(SP)        ;TYPE THE MARKERS
2481  007166  104405                          TYPDS
2482  007170  104401  007176                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2483  007174  000401                          BR      64$             ;;GET OVER THE ASCIZ
2484                                  ;;65$:  .ASCIZ  /-/
2485  007200                          64$:
2436  007200  010001                          MOV     R0,R1           ;FORM THE ADDRESS OF 'SEEK TIME'
2487  007202  006301                          ASL     R1
2488  007204  016103  024010                  MOV     FRWSTR(R1),R3   ;GET THE SEEK TIME
2489  007210  004737  007422                  JSR     PC,PLTPT        ;GO PLOT IT
2490  007214  022700  000004                  CMP     #4,R0           ;PLOTTED UPTO CYL 4?
2491  007220  003402                          BLE     2$              ;YES
2492  007222  005200                          INC     R0              ;INCREMENT BY 1
2493  007224  000755                          BR      1$
```

```
2494  007226  022700  000024          2$:     CMP     #20.,R0         ;PAST CYLINDER 20 ?
2495  007232  003403                  BLE     3$              ;BR IF PAST 20
2496  007234  062700  000002          ADD     #2,R0           ;INCREMENT BY 2
2497  007240  000747                  BR      1$
2498  007242  013746  001236          3$:     MOV     MAXCYL,-(SP)    ;CHECK FOR END
2499  007246  042716  000007          BIC     #7,(SP)         ;CLEAR LOWER CYLINDER BITS
2500  007252  022600                  CMP     (SP)+,R0        ;ALMOST MAX CYL ?
2501  007254  003403                  BLE     4$              ;BR IF ALMOST AT MAX
2502  007256  062700  000005          ADD     #5,R0           ;INCREMENT BY 5
2503  007262  000736                  BR      1$
2504  007264  023700  001236          4$:     CMP     MAXCYL,R0       ;PLOTTED ALL CYLS?
2505  007270  001412                  BEQ     6$              ;BR IF DONE
2506  007272  022737  000312  001236  CMP     #202.,MAXCYL    ;MAXCYL = 202. ?
2507  007300  001003                  BNE     5$              ;BR IF NOT
2508  007302  062700  000002          ADD     #2,R0           ;INCREMENT TO CYLINDER 202
2509  007306  000724                  BR      1$
2510  007310  062700  000005          5$:     ADD     #5,R0           ;INCREMENT TO CYLINDER 405
2511  007314  000721                  BR      1$
2512  007316  000137  007462          6$:     JMP     EXIT12          ;GO TO THE NEXT TEST
2513
2514                                  ;IF SW 4 IS SET THE COMPLETE GRAPH IS PRINTED OUT. IT GIVES TIMES FOR
2515                                  ;SEEKS FROM CYLINDER 0 TO ALL OTHER CYLINDERS (0,1,2,3...MAXCYL).
2516
2517  007322  005000                  CMPGRP: CLR     R0              ;INITIALIZE COUNT
2518  007324  012701  177773                  MOV     #-5,R1          ;INITIALIZE COUNT FOR Y-AXIS MARKER
2519  007330  012702  024010                  MOV     #FRWSTR,R2      ;INITIALIZE POINTER TO SEEK TIMES
2520  007334  104401  001217                  TYPE    ,$CRLF
2521  007340  000404                  BR      3$
2522  007342  005201                  2$:     INC     R1              ;TYPE OUT Y-AXIS MARKER 'CYL #'
2523  007344  001005                  BNE     4$              ;IF REQUIRED
2524  007346  012701  177773          MOV     #-5,R1          ;RESET INDEX
2525  007352  010046                  3$:     MOV     R0,-(SP)        ;TYPE 'CYL #' ON Y-AXIS
2526  007354  104405                  TYPDS                   ;(IN DECIMAL)
2527  007356  000402                  BR      5$
2528  007360  104401  002037          4$:     TYPE    ,BLNKS6         ;6 BLANKS
2529  007364                          5$:
2530  007364  104401  007372                  TYPE    ,6$             ;;TYPE ASCIZ STRING
2531  007370  000401                  BR      6$              ;;GET OVER THE ASCIZ
2532                                  ;;6$:   .ASCIZ  /-/
2533  007374                          6$:
2534  007374  012203                  MOV     (R2)+,R3        ;GET SEEK TIME
2535  007376  004737  007422          JSR     PC,PLTPT        ;GO PLOT THE POINT
2536  007402  104401  001217          TYPE    ,$CRLF
2537  007406  005200                  INC     R0              ;ALL DONE?
2538  007410  022700  001236          CMP     #MAXCYL,R0      ;LAST CYLINDER ?
2539  007414  001352                  BNE     2$              ;IF NOT, GO BACK
2540  007416  000137  007462          6$:     JMP     EXIT12          ;GO TO THE NEXT TEST
2541
2542                                  ;ROUTINE TO PLOT THE SEEK TIME ON THE GRAPH.  ENTER WITH R3 CONTAINING
2543                                  ;THE HORIZONTAL AXIS COORDINATE (I.E. SEEK TIME).
2544
2545  007422  010546                  PLTPT:  MOV     R5,-(SP)        ;STORE R5
2546  007424  012705  000031          MOV     #25.,R5         ;HORIZONTAL AXIS INDEX
2547  007430  162703  000372          1$:     SUB     #250.,R3        ;FIND OUT HOW MANY BLANKS TO
2548  007434  002404                  BLT     2$              ;INSERT TO PLOT THE POINT
2549                                                          ;NOTE THE FIRST CELL = 0 MS
```

```
2550  007436  104401  002044                  TYPE    ,BLNKS1         ;1 BLANK
2551  007442  005305                          DEC     R5              ;AT THE END OF THE HORIZONTAL AXIS ?
2552  007444  100371                          BPL     1$              ;BR IF NOT
2553  007446                          2$:
2554  007446  104401  007454                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2555  007452  000401                          BR      64$             ;;GET OVER THE ASCIZ
2556                                  ;;65$:  .ASCIZ  /X/
2557  007456                          64$:
2558  007456  012605                          MOV     (SP)+,R5        ;RESTORE R5
2559  007460  000207                          RTS     PC
2560
2561  007462  000004                  EXIT12: SCOPE                   ;LOOP ?
2562
2563
2564                                  .SBTTL  END OF PASS ROUTINE
2565
2566                                  ;;**************************************************************************
2567                                  ;*INCREMENT THE PASS NUMBER ($PASS)
2568                                  ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
2569                                  ;*IF THERES A MONITOR GO TO IT
2570                                  ;*IF THERE ISN'T JUMP TO RESTART
2571
2572  007464                          SEOP:
2573  007464  104401  007472                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2574  007470  000410                          BR      64$             ;;GET OVER THE ASCIZ
2575                                  ;;65$:  .ASCIZ  <15><12><12>/END OF PASS/
2576  007512                          64$:
2577  007512  005737  001232                  TST     @#DRVSEL        ;ANY DRIVES SELECTED?
2578  007516  001434                          BEQ     1$              ;NO--BRANCH
2579  007520  104401  007526                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2580  007524  000405                          BR      66$             ;;GET OVER THE ASCIZ
2581                                  ;;67$:  .ASCIZ  / ON DRIVE/
2582  007540                          66$:
2583  007540  013746  001246                  MOV     @#CHKDRV,-(SP)  ;;SAVE @#CHKDRV FOR TYPEOUT
2584  007544  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2585  007546     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
2586  007547     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2587  007550  104401  007556                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2588  007554  000412                          BR      68$             ;;GET OVER THE ASCIZ
2589                                  ;;69$:  .ASCIZ  /   ERRORS DETECTED=/
2590  007602                          68$:
2591  007602  013746  001112                  MOV     @#SERTTL,-(SP)  ;;SAVE @#SERTTL FOR TYPEOUT
2592  007606  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2593  007610  005037  001112          1$:     CLR     @#SERTTL        ;ZERO ERROR TOTAL
2594  007614  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
2595  007620  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
2596  007624  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
2597  007630  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
2598  007636  005327                          DEC     (PC)+           ;;LOOP?
2599  007640  000001          $EOPCT: .WORD   1
2600  007642  003027                          BGT     $DOAGN          ;;YES
2601  007644  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
2602  007646  000001          $ENDCT: .WORD   1
2603  007650  007640                          $EOPCT
2604  007652  104401  007660                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2605  007656  000407                          BR      64$             ;;GET OVER THE ASCIZ
```

```
2606                                        ;;65$:  .ASCIZ  <15><12>/END OF TEST/
2607  007676                                64$:
2608  007676  104401  007726                       TYPE    $ENULL          ;TYPE NULL CHARACTER
2609  007702  013700  000042              $GET42:  MOV     @#42,R0         ;;GET MONITOR ADDRESS
2610  007706  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
2611  007710  000005                              RESET                   ;;CLEAR THE WORLD
2612  007712  004710                      $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
2613  007714  000240                              NOP                     ;;SAVE ROOM
2614  007716  000240                              NOP                     ;;FOR
2615  007720  000240                              NOP                     ;;ACT11
2616  007722                              $DOAGN:
2617  007722  000137                              JMP     @(PC)+          ;;RETURN
2618  007724  003716                      $RTNAD: .WORD   RESTART
2619  007726     377     377     000      $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
2620  007732                                      .EVEN
2621
2622                                        .SBTTL  *** SUBROUTINES ***
2623                                        .SBTTL  ERROR HANDLER ROUTINE
2624
2625                                        ;;**************************************************************
2626                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2627                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2628                                        ;*AND GO TO TYPERR ON ERROR
2629                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2630                                        ;*SW15=1         HALT ON ERROR
2631                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
2632                                        ;*SW10=1         BELL ON ERROR
2633                                        ;*SW09=1         LOOP ON ERROR
2634                                        ;*CALL
2635                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2636
2637  007732                              $ERROR:
2638  007732  104407                              CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2639  007734  105237  001103              7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2640  007740  001775                              BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2641  007742  013777  001102  171172              MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2642  007750  032777  002000  171162              BIT     #BIT10,@SWR     ;;BELL ON ERROR?
2643  007756  001402                              BEQ     1$              ;;NO - SKIP
2644  007760  104401  001212                      TYPE    ,$BELL          ;;RING BELL
2645  007764  005237  001112              1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
2646  007770  011637  001116                      MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2647  007774  162737  000002  001116              SUB     #2,$ERRPC
2648  010002  117737  171110  001114              MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2649  010010  032777  020000  171122              BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2650  010016  001004                              BNE     20$             ;;SKIP TYPEOUTS
2651  010020  004737  010072                      JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
2652  010024  104401  001217                      TYPE    ,$CRLF
2653  010030                              20$:
2654  010030  005777  171104              2$:     TST     @SWR            ;;HALT ON ERROR
2655  010034  100002                              BPL     3$              ;;SKIP IF CONTINUE
2656  010036  000000                              HALT                    ;;HALT ON ERROR!
2657  010040  104407                              CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2658  010042  032777  001000  171070  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
2659  010050  001402                              BEQ     4$              ;;BR IF NO
2660  010052  013716  001110                      MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
2661  010056  005737  001210              4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
```

```
2662  010062  001402                        BEQ     5$                      ;;BR IF NONE
2663  010064  013716  001210                MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
2664  010070                          5$:
2665  010070  000002                        RTI                             ;;RETURN
2666
2667                                  ;;*****************************************************************
2668                                  .SBTTL  TYPERR - TYPE ERROR ROUTINE
2669                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
2670                                  ;*WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
2671                                  ;*TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
2672                                  ;*CONCERNING THE ERROR.
2673                                  ;CALL
2674                                  ;       JSR     PC,@#TYPERR
2675                                  ;       RETURN
2676
2677  010072  104412                  TYPERR: SAVREG                          ;SAVE R0-R5
2678  010074  005037  001204                  CLR     $TMP0                  ;CLEAR LOCATION FOR TEST NUMBER
2679  010100  113737  001102  001204          MOVB    $TSTNM,$TMP0           ;LOAD TEST NUMBER FOR TYPEOUT
2680  010106  005000                          CLR     R0                     ;CLEAR R0 FOR ERROR NUMBER
2681  010110  113700  001114                  MOVB    $ITEMB,R0              ;ERROR NUMBER
2682  010114  005300                          DEC     R0                     ;FORM INDEX FOR ERROR TABLE
2683  010116  006300                          ASL     R0
2684  010120  006300                          ASL     R0
2685  010122  006300                          ASL     R0
2686  010124  062700  002046          1$:     ADD     #$ERRTB,R0             ;FORM ADDRESS
2687  010130  012037  010144                  MOV     (R0)+,2$               ;GET ERROR MESSAGE (EM) POINTER
2688  010134  001412                          BEQ     3$                     ;BRANCH IF THERE ISN'T ONE
2689  010136  104401  001217                  TYPE    ,$CRLF                 ;"CARRIAGE RETURN - LINE FEED
2690  010142  104401                          TYPE
2691  010144  000000                  2$:     .WORD   0                      ;"EM" POINTER GOES HERE
2692  010146  032777  000100  170764          BIT     #SW06,@SWR             ;TYPE THE REGISTERS INSTEAD OF THE NORMAL
2693                                                                         ;'DH' & 'DT' LINES ?
2694  010154  001402                          BEQ     3$                     ;BR IF NOT
2695  010156  012700  023502                  MOV     #ETSP,R0               ;CHANGE MESSAGE BLOCK
2696  010162  012037  010176          3$:     MOV     (R0)+,4$               ;PICK UP DATA HEADER (DH) POINTER
2697  010166  001404                          BEQ     5$                     ;BRANCH IF NONE
2698  010170  104401  001217                  TYPE    ,$CRLF                 ;CARRIAGE RETURN-LINE FEED
2699  010174  104401                          TYPE
2700  010176  000000                  4$:     .WORD   0                      ;"DH" POINTER GOES HERE
2701  010200  012001                  5$:     MOV     (R0)+,R1               ;PICKUP DATA TABLE (DT) POINTER
2702  010202  001450                          BEQ     20$                    ;BRANCH IF NONE
2703  010204  005005                          CLR     R5                     ;SET INDENT SWITCH
2704  010206  012000                          MOV     (R0)+,R0               ;DATA FORMAT (DF) POINTER
2705  010210  012002                          MOV     (R0)+,R2               ;NUMBER OF DH'S TO TYPE
2706  010212  001441                          BEQ     17$                    ;BRANCH IF DH NUMBER IS 0
2707  010214  005105                          COM     R5                     ;NO INDENT
2708  010216  104401  001217                  TYPE    ,$CRLF                 ;CARRIAGE RETURN-LINE FEED
2709  010222  112003                  10$:    MOVB    (R0)+,R3               ;NUMBER OF DATA WORDS TO TYPE
2710  010224  112004                          MOVB    (R0)+,R4               ;AND HOW TO TYPE THEM
2711  010226  006004                  11$:    ROR     R4                     ;OCTAL OR DECIMAL?
2712  010230  103403                          BCS     12$                    ;DECIMAL--BRANCH
2713  010232  013146                          MOV     @(R1)+,-(SP)           ;;SAVE @(R1)+ FOR TYPEOUT
2714  010234  104402                          TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2715  010236  000402                          BR      13$
2716  010240                          12$:
2717  010240  013146                          MOV     @(R1)+,-(SP)           ;;SAVE @(R1)+ FOR TYPEOUT
```

```
2718  010242  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2719  010244  005303                13$:     DEC     R3              ;MORE NUMBERS TO TYPE?
2720  010246  001403                        BEQ     14$             ;NO--BRANCH
2721  010250  104401  002043                TYPE    ,BLNKS2         ;YES--TYPE SEPARATORS
2722  010254  000764                        BR      11$             ;LOOP
2723  010256  005302                14$:     DEC     R2              ;MORE DH'S?
2724  010260  003421                        BLE     20$             ;NO--BRANCH
2725  010262  104401  001217                TYPE    ,$CRLF          ;YES--START A NEW LINE
2726  010266  005105                        COM     R5              ;INDENT?
2727  010270  001002                        BNE     15$             ;NO--BRANCH
2728  010272  104401  002043                TYPE    ,BLNKS2         ;YES--TYPE SPACES
2729  010276  012037  010304        15$:    MOV     (R0)+,16$       ;GET NEXT DH
2730  010302  104401                        TYPE                    ;AND TYPE IT
2731  010304  000000                16$:     .WORD   0               ;DH POINTER GOES HERE
2732  010306  104401  001217                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
2733  010312  005705                        TST     R5              ;INDENT?
2734  010314  001342                        BNE     10$             ;NO--BRANCH
2735  010316  104401  002043        17$:    TYPE    ,BLNKS2         ;YES--TYPE SPACES
2736  010322  000737                        BR      10$             ;LOOP
2737  010324  104413                20$:     RESREG                  ;RESTORE R0-R5
2738  010326  000207                        RTS     PC              ;RETURN
2739
2740
2741
2742                                  .SBTTL  TYPE ROUTINE
2743
2744                                  ;;**************************************************************
2745                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2746                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2747                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2748                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2749                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2750                                  ;*
2751                                  ;*CALL:
2752                                  ;*1) USING A TRAP INSTRUCTION
2753                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2754                                  ;*OR
2755                                  ;*      TYPE
2756                                  ;*      MESADR
2757                                  ;*
2758
2759  010330  105737  001157  .       $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2760  010334  100002                        BPL     1$              ;;BR IF YES
2761  010336  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
2762  010340  000407                        BR      3$              ;;LEAVE
2763  010342  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
2764  010344  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2765  010350  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2766  010352  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2767  010354  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2768  010356  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
2769  010360  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2770  010364  000002                        RTI                     ;;RETURN
2771  010366  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF (HT)
2772  010372  001430                        BEQ     8$
2773  010374  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT (CRLF)
```

```
2774  010400  001006                        BNE     5$
2775  010402  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
2776  010404  104401                        TYPE                    ;;TYPE A CR AND LF
2777  010406  001217                        $CRLF
2778  010410  105037  010544                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2779  010414  000755                        BR      2$              ;;GET NEXT CHARACTER
2780  010416  004737  010500        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2781  010422  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2782  010426  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
2783  010430  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2784                                                                ;;AND THE NULL CHAR.
2785  010434  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2786  010440  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2787  010442  004737  010500                JSR     PC,$TYPEC       ;;GO TYPE A NULL
2788  010446  105337  010544                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2789  010452  000770                        BR      7$              ;;LOOP
2790
2791                                  ;HORIZONTAL TAB PROCESSOR
2792
2793  010454  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2794  010460  004737  010500        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2795  010464  132737  000007  010544        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2796  010472  001372                        BNE     9$              ;;TAB STOP
2797  010474  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
2798  010476  000724                        BR      2$              ;;GET NEXT CHARACTER
2799  010500  105777  170444        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2800  010504  100375                        BPL     $TYPEC
2801  010506  116677  000002  170436        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2802  010514  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2803  010522  001003                        BNE     1$              ;;BRANCH IF NO
2804  010524  105037  010544                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2805  010530  000406                        BR      $TYPEX          ;;EXIT
2806  010532  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2807  010540  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
2808  010542  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
2809  010544  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
2810  010546  000207          $TYPEX: RTS     PC
2811
2812
2813                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2814
2815                                  ;;**************************************************************
2816                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2817                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2818                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2819                                  ;*CALL:
2820                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2821                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
2822                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2823                                  ;*      .BYTE   M               ;;M=1 OR 0
2824                                  ;*                                      ;;1=TYPE LEADING ZEROS
2825                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
2826                                  ;*
2827                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2828                                  ;*$TYPOS OR $TYPOC
2829                                  ;*CALL:
```

```
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                        ;*
010550  017646  000000          STYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
010554  116637  000001  010773          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
010562  112637  010775                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
010566  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
010572  000406                          BR      $TYPON
010574  112737  000001  010773  STYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
010602  112737  000006  010775          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
010610  112737  000005  010772  STYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
010616  010346                          MOV     R3,-(SP)        ;;SAVE R3
010620  010446                          MOV     R4,-(SP)        ;;SAVE R4
010622  010546                          MOV     R5,-(SP)        ;;SAVE R5
010624  113704  010775                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
010630  005404                          NEG     R4
010632  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
010636  110437  010774                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
010642  113704  010773                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
010646  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
010652  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
010654  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
010656  000404                          BR      3$              ;;GO DO MSB
010660  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
010662  006105                          ROL     R5
010664  006105                          ROL     R5
010666  010503                          MOV     R5,R3
010670  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
010672  105337  010774                  DECB    $OMODE          ;;TYPE THIS DIGIT?
010676  100016                          BPL     7$              ;;BR IF NO
010700  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
010704  001002                          BNE     4$              ;;TEST FOR 0
010706  005704                          TST     R4              ;;SUPPRESS THIS 0?
010710  001403                          BEQ     5$              ;;BR IF YES
010712  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
010714  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
010720  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
010724  110337  010770                  MOVB    R3,8$           ;;SAVE FOR TYPING
010730  104401  010770                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
010734  105337  010772          7$:     DECB    $OCNT           ;;COUNT BY 1
010740  003347                          BGT     2$              ;;BR IF MORE TO DO
010742  002402                          BLT     6$              ;;BR IF DONE
010744  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
010746  000744                          BR      2$              ;;GO DO THE LAST DIGIT
010750  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
010752  012604                          MOV     (SP)+,R4        ;;RESTORE R4
010754  012603                          MOV     (SP)+,R3        ;;RESTORE R3
010756  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
010764  012616                          MOV     (SP)+,(SP)
010766  000002                          RTI                     ;;RETURN
010770      000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
```

```
2896  010771  000                     .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2897  010772  000             $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2898  010773  000             $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2899  010774  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2900
2901                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2902
2903                          ;;**************************************************************
2904                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2905                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2906                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2907                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2908                          ;*REPLACED WITH SPACES.
2909                          ;*CALL:
2910                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2911                          ;*      TYPOS                   ;;GO TO THE ROUTINE
2912
2913  010776                  $TYPOS:
2914  010776  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2915  011000  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2916  011002  010246                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2917  011004  010346                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2918  011006  010546                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2919  011010  012746  020200         MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2920  011014  016605  000020         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
2921  011020  100004                 BPL     1$              ;;BR IF INPUT IS POS.
2922  011022  005405                 NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2923  011024  112766  000055  000001 MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2924  011032  005000          1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
2925  011034  012703  011212         MOV     #$OBLK,R3       ;;SETUP THE OUTPUT POINTER
2926  011040  112723  000040         MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2927  011044  005002          2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
2928  011046  016001  011202         MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2929  011052  160105          3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
2930  011054  002402                 BLT     4$              ;;BR IF DONE
2931  011056  005202                 INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2932  011060  000774                 BR      3$
2933  011062  060105          4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
2934  011064  005702                 TST     R2              ;;CHECK IF BCD DIGIT=0
2935  011066  001002                 BNE     5$              ;;FALL THROUGH IF 0
2936  011070  105716                 TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2937  011072  100407                 BMI     7$              ;;BR IF YES
2938  011074  106316          5$:    ASLB    (SP)            ;;MSD?
2939  011076  103003                 BCC     6$              ;;BR IF NO
2940  011100  116663  000001 177777 MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2941  011106  052702  000060  6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2942  011112  052702  000040  7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2943  011116  110223                 MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2944  011120  005720                 TST     (R0)+           ;;JUST INCREMENTING
2945  011122  020027  000010         CMP     R0,#10          ;;CHECK THE TABLE INDEX
2946  011126  002746                 BLT     2$              ;;GO DO THE NEXT DIGIT
2947  011130  003002                 BGT     8$              ;;GO TO EXIT
2948  011132  010502                 MOV     R5,R2           ;;GET THE LSD
2949  011134  000764                 BR      6$              ;;GO CHANGE TO ASCII
2950  011136  105726          8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2951  011140  100003                 BPL     9$              ;;BR IF NO
```

```
 2942  011142  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 2943  011150  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 2944  011152  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 2945  011154  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 2946  011156  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 2947  011160  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 2948  011162  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 2949  011164  104401  011212                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 2950  011170  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 2951  011176  012616                          MOV     (SP)+,(SP)
 2952  011200  000002                          RTI                     ;;RETURN TO USER
 2953  011202  023420                  $DTBL:  10000.
 2954  011204  001750                          1000.
 2955  011206  000144                          100.
 2956  011210  000012                          10.
 2957  011212  000004                  $DBLK:  .BLKW   4
 2958
 2959                                  .SBTTL  TTY INPUT ROUTINE
 2960
 2961                                  ;;**************************************************************
 2962                                  .ENABL  LSB
 2963  011222  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
 2964  011224  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
 2965  011226  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
 2966  011230  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
 2967          011231                  $TKQEND=.
 2968          011232                  .EVEN
 2969
 2970                                  ;*TK INITIALIZE ROUTINE
 2971                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 2972                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 2973                                  ;
 2974                                  ;*CALL:
 2975                                  ;*      JSR     PC,$TKINT
 2976                                  ;*      RETURN
 2977                                  ;
 2978  011232  005037  011222          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 2979  011236  012737  011230  011224          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 2980  011244  013737  011224  011226          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 2981  011252  012737  011302  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 2982  011260  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 2983  011266  005777  167654                  TST     @$TKB           ;;CLEAR DONE FLAG
 2984  011272  012777  000100  167644          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 2985  011300  000207                          RTS     PC              ;;RETURN TO CALLER
 2986
 2987                                  ;*TK SERVICE ROUTINE
 2988                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 2989                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 2990                                  ;*IT IN THE QUEUE.
 2991                                  ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
 2992                                  ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS ($START)
 2993                                  ;
 2994  011302  117746  167640          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 2995  011306  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
 2996  011312  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
 2997  011316  001007                          BNE     1$              ;;BRANCH IF NO
```

```
2998  011320  104401  012441                TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
2999  011324  004737  011232                JSR     PC,$TKINT       ;;INIT THE KEYBOARD
3000  011330  005726                        TST     (SP)+           ;;CLEAN UP STACK
3001  011332  000137  002710                JMP     START5          ;;CONTROL C RESTART
3002  011336  021627  000007        1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
3003  011342  001004                        BNE     2$              ;;BRANCH IF NO
3004  011344  022737  000176  001140        CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
3005  011352  001500                        BEQ     6$              ;;GO TO SWR CHANGE
3006
3007  011354                        2$:
3008  011354  022737  000001  011222        CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
3009  011362  001004                        BNE     3$              ;;BRANCH IF NO
3010  011364  104401  001212                TYPE    ,$BELL          ;;RING THE TTY BELL
3011  011370  005726                        TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
3012  011372  000451                        BR      5$              ;;EXIT
3013  011374  021627  000023        3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
3014  011400  001021                        BNE     32$             ;;BRANCH IF NO
3015  011402  005077  167536                CLR     @#TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
3016  011406  005726                        TST     (SP)+           ;;CLEAN CHAR OFF STACK
3017  011410  105777  167530        31$:    TSTB    @#TKS           ;;WAIT FOR A CHAR
3018  011414  100375                        BPL     31$             ;;LOOP UNTIL ITS THERE
3019  011416  117746  167524                MOVB    @#TKB,-(SP)     ;;GET THE CHARACTER
3020  011422  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3021  011426  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3022  011432  001366                        BNE     31$             ;;BRANCH IF NO
3023  011434  012777  000100  167502        MOV     #100,@#TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
3024  011442  000002                        RTI                     ;;RETURN
3025  011444  005237  011222        32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
3026  011450  021627  000140                CMP     (SP),#140       ;;IS IT UPPER CASE?
3027  011454  002405                        BLT     4$              ;;BRANCH IF YES
3028  011456  021627  000175                CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
3029  011462  003002                        BGT     4$              ;;BRANCH IF YES
3030  011464  042716  000040                BIC     #40,(SP)        ;;MAKE IT UPPER CASE
3031  011470  112677  177530        4$:     MOVB    (SP)+,@#TKQIN   ;;AND PUT IT IN QUEUE
3032  011474  005237  011224                INC     $TKQIN          ;;UPDATE THE POINTER
3033  011500  023727  011224  011231        CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
3034  011506  001003                        BNE     5$              ;;BRANCH IF NO
3035  011510  012737  011230  011224        MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
3036  011516  000002                5$:     RTI                     ;;RETURN
3037
3038                                  ;;**********************************************************************
3039                                  ;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3040                                  ;;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
3041                                  ;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
3042                                  ;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
3043  011520  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
3044  011526  001124                        BNE     15$             ;;EXIT IF NOT
3045  011530  105777  167410                TSTB    @#TKS           ;;IS A CHAR WAITING?
3046  011534  100121                        BPL     15$             ;;IF NOT, EXIT
3047  011536  117746  167404                MOVB    @#TKB,-(SP)     ;;YES
3048  011542  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
3049  011546  021627  000007                CMP     (SP),#7         ;;IS IT A CONTROL-G?
3050  011552  001300                        BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
3051                                                                ;;AND EXIT
3052
3053                                  ;;**********************************************************************
```

```
3054                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
3055                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
3056                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
3057  011554  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
3058  011562  001674                          BEQ     2$              ;;BRANCH IF YES
3059  011564  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
3060  011566  004737  011232                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
3061  011572  005077  167346                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
3062  011576  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
3063
3064  011604  104401  012453          $GTSWR: TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (↑G)
3065  011610  104401  012460                  TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
3066  011614  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE $WREG FOR TYPEOUT
3067  011620  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3068  011622  104401  012471                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
3069  011626  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
3070  011630  005046                          CLR     -(SP)           ;;THE NEW SWR
3071  011632  105777  167306          7$:     TSTB    @$TKS           ;;CHAR THERE?
3072  011636  100375                          BPL     7$              ;;IF NOT TRY AGAIN
3073
3074  011640  117746  167302                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
3075  011644  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3076
3077  011650  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
3078  011654  001015                          BNE     9$              ;;BRANCH IF NOT
3079  011656  104401  012441                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
3080  011662  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
3081  011666  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
3082  011674  001003                          BNE     8$              ;;BRANCH IF NO
3083  011676  012777  000100  167240          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
3084  011704  000137  002710          8$:     JMP     START$          ;;CONTROL-C RESTART
3085
3086
3087  011710  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
3088  011714  001005                          BNE     10$             ;;BRANCH IF NOT
3089  011716  104401  012445                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
3090  011722  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
3091  011726  000737                          BR      19$             ;;LET'S TRY IT AGAIN
3092
3093
3094  011730  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
3095  011734  001022                          BNE     16$             ;;BRANCH IF NO
3096  011736  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
3097  011742  001403                          BEQ     11$             ;;BRANCH IF YES
3098  011744  016677  000002  167166          MOV     2(SP),@SWR      ;;SAVE NEW SWR
3099  011752  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
3100  011756  104401  001217          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
3101  011762  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
3102  011770  001003                          BNE     15$             ;;BRANCH IF NOT
3103  011772  012777  000100  167144          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
3104  012000  000002                  15$:    RTI                     ;;RETURN
3105  012002  004737  010500          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
3106  012006  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
3107  012012  002420                          BLT     18$             ;;BRANCH IF YES
3108  012014  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
3109  012020  003015                          BGT     18$             ;;BRANCH IF YES
```

```
3110  012022  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3111  012026  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
3112  012032  001403                        BEQ     17$             ;;BRANCH IF YES
3113  012034  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
3114  012036  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
3115  012040  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
3116  012042  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3117  012046  056615  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3118  012052  000667                        BR      7$              ;;GET THE NEXT ONE
3119  012054  104401  001216        18$:    TYPE    ,$QUES          ;;TYPE ?(CR)(LF)
3120  012060  000720                        BR      20$             ;;SIMULATE CONTROL-U
3121                                .DSABL  LSB
3122
3123
3124                                ;;*****************************************************************
3125                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3126                                ;*CALL:
3127                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
3128                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3129                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
3130                                ;
3131
3132  012062  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
3133  012064  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE PS
3134  012072  005066  000004                CLR     4(SP)           ;;GET READY FOR A CHARACTER
3135  012076  005046                        CLR     -(SP)           ;;PUT NEW PS ON STACK
3136  012100  012746  012106                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3137  012104  000002                        RTI                     ;;POP NEW PC AND PS
3138  012106                        64$:
3139  012106  005737  011222        1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
3140  012112  001775                        BEQ     1$
3141  012114  005337  011222                DEC     $TKCNT          ;;DECREMENT THE COUNTER
3142  012120  117766  177102  000004        MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
3143  012126  005237  011226                INC     $TKQOUT         ;;UPDATE THE POINTER
3144  012132  023727  011226  011231        CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
3145  012140  001003                        BNE     2$              ;;BRANCH IF NO
3146  012142  012737  011230  011226        MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
3147  012150  000002                2$:     RTI                     ;;RETURN
3148                                ;;*****************************************************************
3149                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3150                                ;*CALL:
3151                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
3152                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3153                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
3154
3155  012152  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
3156  012154  005046                        CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
3157  012156  012703  012406        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
3158  012162  022703  012441        2$:     CMP     #$TTYIN+27.,R3  ;;BUFFER FULL?
3159  012166  101456                        BLOS    4$              ;;BR IF YES
3160  012170  104410                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
3161  012172  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
3162  012174  122713  000177        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
3163  012200  001022                        BNE     5$              ;;BR IF NO
3164  012202  005716                        TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
3165  012204  001007                        BNE     6$              ;;BR IF NO
```

```
3166  012206  112737  000134  012404          MOVB    #'\,@$      ;;TYPE A BACK SLASH
3167  012214  104401  012404                  TYPE    @$
3168  012220  012716  177777                  MOV     #-1,(SP)    ;;SET THE RUBOUT KEY
3169  012224  005303                  6$:     DEC     R3          ;;BACKUP BY ONE
3170  012226  020327  012406                  CMP     R3,#STTYIN  ;;STACK EMPTY?
3171  012232  103434                  BLO     4$          ;;BR IF YES
3172  012234  111337  012404                  MOVB    (R3),@$     ;;SETUP TO TYPEOUT THE DELETED CHAR.
3173  012240  104401  012404                  TYPE    @$          ;;GO TYPE
3174  012244  000746                  BR      2$          ;;GO READ ANOTHER CHAR.
3175  012246  005716                  5$:     TST     (SP)        ;;RUBOUT KEY SET?
3176  012250  001406                  BEQ     7$          ;;BR IF NO
3177  012252  112737  000134  012404          MOVB    #'\,@$      ;;TYPE A BACK SLASH
3178  012260  104401  012404                  TYPE    @$
3179  012264  005016                  CLR     @SP)        ;;CLEAR THE RUBOUT KEY
3180  012266  122713  000025          7$:     CMPB    #25,(R3)    ;;IS CHARACTER A CTRL U?
3181  012272  001003                  BNE     8$          ;;BR IF NO
3182  012274  104401  012446                  TYPE    @CNTLU      ;;TYPE A CONTROL "U"
3183  012300  000726                  BR      1$          ;;GO START OVER
3184  012302  122713  000022          8$:     CMPB    #22,(R3)    ;;IS CHARACTER A "↑R"?
3185  012306  001011                  BNE     3$          ;;BRANCH IF NO
3186  012310  105013                  CLRB    (R3)        ;;CLEAR THE CHARACTER
3187  012312  104401  001217                  TYPE    @CRLF       ;;TYPE A "CR" & "LF"
3188  012316  104401  012406                  TYPE    @STTYIN     ;;TYPE THE INPUT STRING
3189  012322  000717                  BR      2$          ;;GO PICKUP ANOTHER CHACTER
3190  012324  104401  001216          4$:     TYPE    @QUES       ;;TYPE A '?'
3191  012330  000712                  BR      1$          ;;CLEAR THE BUFFER AND LOOP
3192  012332  111337  012404          3$:     MOVB    (R3),@$     ;;ECHO THE CHARACTER
3193  012336  104401  012404                  TYPE    @$
3194  012342  122723  000015                  CMPB    #15,(R3)+   ;;CHECK FOR RETURN
3195  012346  001305                  BNE     2$          ;;LOOP IF NOT RETURN
3196  012350  105063  177777                  CLRB    -1(R3)      ;;CLEAR RETURN (THE 15)
3197  012354  104401  001220                  TYPE    @LF         ;;TYPE A LINE FEED
3198  012360  005726                  TST     @SP)+       ;;CLEAN RUBOUT KEY FROM THE STACK
3199  012362  012603                  MOV     (SP)+,R3    ;;RESTORE R3
3200  012364  011646                  MOV     (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
3201  012366  016666  000004  000002          MOV     4(SP),2(SP) ;;       FIRST ASCII CHARACTER ON IT
3202  012374  012766  012406  000004          MOV     #STTYIN,4(SP)
3203  012402  000002                  RTI                 ;;RETURN
3204  012404     000          9$:     .BYTE   0           ;;STORAGE FOR ASCII CHAR. TO TYPE
3205  012405     000                  .BYTE   0           ;;TERMINATOR
3206  012406  000033          STTYIN: .BLKB   27.         ;;RESERVE 27. BYTES FOR TTY INPUT
3207  012441     136  006503  000012 $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
3208  012446  052536  005015     000 $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
3209  012453     136  006507  000012 $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
3210  012460  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
3211  012466  020075     000
3212  012471     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
3213  012476  036440  000040
3214
3215                                  .SBTTL  SCOPE HANDLER ROUTINE
3216
3217                                  ;;**************************************************************
3218                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3219                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3220                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3221                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
```

```
                                        ;*SW14=1        LOOP ON TEST
                                        ;*SW11=1        INHIBIT ITERATIONS
                                        ;*SW09=1        LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
                                        $SCOPE:
012502                                                  CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
012502  104407                                        1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
012504  032777  040000  166426                        BNE     $OVER           ;;YES IF SW14=1
012512  001076                                        ;*****START OF CODE FOR THE XOR TESTER*****
                                        $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
012514  000416                                                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
012516  013746  000004                                        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
012522  012737  012542  000004                            TST     @#177060        ;;TIME OUT ON XOR?
012530  005737  177060                                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
012534  012637  000004                                    BR      $SVLAD          ;;GO TO THE NEXT TEST
012540  000450                                        5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
012542  022626                                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
012544  012637  000004                                    BR      7$              ;;LOOP ON THE PRESENT TEST
012550  000413                                        6$:;*****END OF CODE FOR THE XOR TESTER*****
012552                                                  TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
012554  105737  001103                                    BEQ     3$              ;;BR IF NO
012560  001421                                        CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
012562  123737  001115  001103                            BHI     3$              ;;BR IF NO
012570  101015                                        BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
012572  032777  001000  166342                            BEQ     4$              ;;BR IF NO
012600  001404                                        MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
012602  013737  001110  001106                        BR      $OVER
012610  000440                                        CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
012614  105037  001103                                4$:     CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
012620  005037  001206                                    BR      1$              ;;ESCAPE TO THE NEXT TEST
012622  000412                                        BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
012630  032777  004000  166310                        3$:     BNE     1$              ;;BR IF YES
012632  001006                                        INC     $ICNT           ;;INCREMENT ITERATION COUNT
012636  005237  001104                                    CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
012644  023737  001206  001104                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
012646  002021                                        MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
012654  012737  000001  001104                        1$:     MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
012662  012737  012724  001206                        $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
012666  105237  001102                                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
012672  011637  001106                                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
012676  011637  001110                                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
012702  005037  001210                                    MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERR R ON NEXT TEST
012710  112737  000001  001115                        $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
012716  013777  001102  166224                            MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
012722  013716  001106                                    RTI                     ;;FIXES PS
012724  000002                                        $MXCNT: 1               ;;MAX. NUMBER OF ITERATIONS
000001

                                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                        ;;**************************************************************
                                        ;*SAVE R0-R5
                                        ;*CALL:
                                        ;*      SAVREG
```

```
                                        ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                        ;*
                                        ;*TOP---(+16)
                                        ;* +2---(+18)
                                        ;* +4---R5
                                        ;* +6---R4
                                        ;* +8---R3
                                        ;*+10---R2
                                        ;*+12---R1
                                        ;*+14---R0

012726                                  $SAVREG:
012726  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012730  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012732  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012734  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
012736  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
012740  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
012742  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
012746  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
012752  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
012756  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
012762  000002                                  RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
012764                                  $RESREG:
012764  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
012770  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
012774  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
013000  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
013004  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
013006  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
013010  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
013012  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
013014  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
013016  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
013020  000002                                  RTI

                                        .SBTTL  INTEGER MULTIPLY ROUTINE

                                        ;;*****************************************************************
                                        ;*CALL
                                        ;*      MOV     MULTIPLER,-(SP)
                                        ;*      MOV     MULTIPLICAND,-(SP)
                                        ;*      JSR     PC,@#$MULT
                                        ;*      RETURN  ;;PRODUCT IS ON THE STACK
                                        ;*
                                        ;*      STACK   PRODUCT
                                        ;*      -----   -------
                                        ;*      TOP     LSB'S
                                        ;*      +2      MSB'S

013022                                  $MULT:
013022  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

```
 3334  013024  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 3335  013026  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 3336  013030  005046                        CLR     -(SP)           ;;CLEAR THE SIGN KEY
 3337  013032  016601  000012                MOV     12(SP),R1       ;;GET THE MULTIPLICAND
 3338  013036  100002                        BPL     1$              ;;BR IF PLUS
 3339  013040  005216                        INC     (SP)            ;;SET THE SIGN KEY
 3340  013042  005401                        NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
 3341  013044  016602  000014        1$:     MOV     14(SP),R2       ;;GET THE MULTIPLIER
 3342  013050  100002                        BPL     2$              ;;BR IF PLUS
 3343  013052  005316                        DEC     (SP)            ;;UPDATE THE SIGN KEY
 3344  013054  005402                        NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
 3345  013056  012746  000021        2$:     MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
 3346  013062  005000                        CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
 3347  013064  103001                3$:     BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
 3348  013066  060200                        ADD     R2,R0
 3349  013070  006000                4$:     ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
 3350  013072  006001                        ROR     R1              ;;THE MULTIPLICAND
 3351  013074  005316                        DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
 3352  013076  001372                        BNE     3$              ;;BR IF NO
 3353  013100  022616                        CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
 3354  013102  001403                        BEQ     5$              ;;GO TO EXIT IF NO
 3355  013104  005400                        NEG     R0              ;;YES--SO MAKE IT SO
 3356  013106  005401                        NEG     R1
 3357  013110  005600                        SBC     R0
 3358  013112  005726                5$:     TST     (SP)+           ;;CLEAR SIGN INFO. OFF OF STACK
 3359  013114  010066  000012              MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S)
 3360  013120  010166  000010              MOV     R1,10(SP)       ;;LSB'S
 3361  013124  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 3362  013126  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 3363  013130  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 3364  013132  000207                        RTS     PC
 3365
 3366                                  .SBTTL  TRAP DECODER
 3367
 3368                                  ;;**************************************************************
 3369                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 3370                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 3371                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 3372                                  ;*GO TO THAT ROUTINE.
 3373
 3374  013134  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 3375  013136  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
 3376  013142  005740                        TST     -(R0)           ;;BACKUP BY 2
 3377  013144  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 3378  013146  006300                        ASL     R0              ;;POSITION FOR INDEXING
 3379  013150  016000  013170              MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 3380  013154  000200                        RTS     R0              ;;GO TO ROUTINE
 3381
 3382
 3383                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
 3384
 3385  013156  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
 3386  013160  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
 3387  013166  000002                        RTI                     ;;RESTORE THE PSW
 3388
 3389                                  .SBTTL  TRAP TABLE
```

```
                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE "TRAP" INSTRUCTION.
                                        ;
                                        ;       ROUTINE
                                        ;       -------
013170  013156                  $TRPAD: .WORD   $TRAP2
013172  010330                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
013174  010574                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
013176  010550                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
013200  010610                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
013202  010776                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

013204  011610                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

013206  011520                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
013210  012062                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
013212  012152                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
013214  012726                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
013216  012764                          $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE

                                        .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                        ;*CALL
                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                        ;*      JSR     PC,@#$$B2D      ;;CALL
                                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK

013220  016637  000002  013250   $$B2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
013226  012746  013250                   MOV     #1$,-(SP)       ;;SET POINTER
013232  004737  013254                   JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
013236  062716  000005                   ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
013242  012666  000002                   MOV     (SP)+,2(SP)     ;;PICKUP POINTER
013246  000207                           RTS     PC              ;;RETURN
013250  000000  000000           1$:     .WORD   0,0

                                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                        ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                        ;*POSITIVE.
                                        ;*CALL
                                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                        ;*      JSR     PC,@#$DB2D
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK

013254  104412                   $DB2D:  SAVREG                  ;;SAVE REGISTERS
013256  016602  000002                   MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
013262  012700  013434                   MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
```

```
3446  013266  010066  000002                    MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
3447  013272  012201                            MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
3448  013274  012202                            MOV     (R2)+,R2
3449  013276  012737  000012  013352            MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
3450  013304  012704  013364                    MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
3451  013310  012705  013366                    MOV     #$TNPWR+2,R5
3452  013314  005003                    1$:     CLR     R3              ;;CLEAR PARTIAL
3453  013316  161401                    2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
3454  013320  005602                            SBC     R2
3455  013322  161502                            SUB     (R5),R2
3456  013324  002402                            BLT     3$              ;;BR IF TEN POWER TO LARGE
3457  013326  005203                            INC     R3              ;;ADD 1 TO PARTIAL
3458  013330  000772                            BR      2$              ;;LOOP
3459  013332  062401                    3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
3460  013334  005502                            ADC     R2
3461  013336  062402                            ADD     (R4)+,R2
3462  013340  022525                            CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
3463  013342  052703  000060                    BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
3464  013346  110320                            MOVB    R3,(R0)+        ;;SAVE IT
3465  013350  005327                            DEC     (PC)+           ;;DONE?
3466  013352  000000                    4$:     .WORD   0
3467  013354  001357                            BNE     1$              ;;BR IF NO
3468  013356  105020                            CLRB    (R0)+           ;;TERMINATOR
3469  013360  104413                            RESREG                  ;;RESTORE REGISTERS
3470  013362  000207                            RTS     PC              ;;RETURN
3471  013364  145000                    $TNPWR: 145000                  ;;1.0E09
3472  013366  035632                            35632
3473  013370  160400                            160400                  ;;1.0E08
3474  013372  002765                            2765
3475  013374  113200                            113200                  ;;1.0E07
3476  013376  000230                            230
3477  013400  041100                            041100                  ;;1.0E06
3478  013402  000017                            17
3479  013404  103240                            103240                  ;;1.0E05
3480  013406  000001                            1
3481  013410  023420                            23420                   ;;1.0E04
3482  013412  000000                            0
3483  013414  001750                            1750                    ;;1.0E03
3484  013416  000000                            0
3485  013420  000144                            144                     ;;1.0E02
3486  013422  000000                            0
3487  013424  000012                            12                      ;;1.0E01
3488  013426  000000                            0
3489  013430  000001                            1                       ;;1.0E00
3490  013432  000000                            0
3491  013434  000014                    $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
3492
3493                                    .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
3494
3495                                    ;;**************************************************************
3496                                    ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
3497                                    ;*LEADING NUMBERS.
3498                                    ;*CALL
3499                                    ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
3500                                    ;*      JSR     PC,@#$SUPRS
3501
```

```
3502
3503  013450  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
3504  013452  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
3505  013456  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
3506  013460  001403                          BEQ     2$              ;;BR IF YES
3507  013462  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
3508  013466  001773                          BEQ     1$              ;;BR IF YES
3509  013470  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
3510  013472  010037  013500                  MOV     R0,3$           ;;SAVE FOR TYPING
3511  013476  104401                          TYPE                    ;;GO TYPE
3512  013500  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
3513  013502  012600                          MOV     (SP)+,R0        ;;RESTORE R0
3514  013504  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
3515  013506  000207                          RTS     PC              ;;RETURN
3516
3517                                  .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
3518
3519                                  ;;**************************************************************
3520                                  ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
3521                                  ;*WITH A RANGE OF 0 TO 2(+33)-1.
3522                                  ;*CALL:
3523                                  ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
3524                                  ;*      RETURN                  ;;RETURN HERE THE RANDOM
3525                                  ;*                              ;;NUMBER WILL BE IN
3526                                  ;*                              ;;$HINUM,$LONUM
3527
3528  013510                          $RAND:
3529  013510  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3530  013512  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3531  013514  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3532  013516  013700  013610                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
3533  013522  013701  013606                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
3534  013526  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
3535  013532  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
3536  013534  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
3537  013536  005202                          INC     R2              ;;CHECK FOR DONE
3538  013540  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
3539  013542  063700  013610                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
3540  013546  005501                          ADC     R1              ;;PROPOGATE CARRY
3541  013550  063701  013606                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
3542  013554  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
3543  013560  005501                          ADC     R1              ;;PROPOGATE CARRY
3544  013562  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
3545  013566  010037  013610                  MOV     R0,$LONUM       ;;SAVE R0
3546  013572  010137  013606                  MOV     R1,$HINUM       ;;SAVE R1
3547  013576  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3548  013600  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3549  013602  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3550  013604  000207                          RTS     PC              ;;RETURN
3551  013606  176543                  $HINUM: .WORD   176543
3552  013610  123456                  $LONUM: .WORD   123456
3553
3554                                  ;;**************************************************************
3555                                  .SBTTL  RPINIT - INITIALIZE THE RP11 & SEE WHICH DRIVES ARE AVAILABLE
3556                                  ;*THIS ROUTINE CLEARS THE RP11 AND DETERMINES WHICH DRIVES ARE AVAILABLE.
3557                                  ;*THE TABLE 'DRVSTA' IS LOADED TO REFLECT THE SYSTEM STATUS.
```

```
                                        ;CALL
                                        ;       JSR     PC,@#RPINIT
                                        ;       RETURN
                                        RPINIT: SAVREG                  ;SAVE R0-R5
013612  104412                                  MOV     RPVEC,R1        ;VECTOR ADDRESS
013614  013701  001270                          CLR     (R1)+           ;SET INTERRUPT ADDRESS TO ZERO
013620  005021                                  MOV     RPPRIO,(R1)     ;RP11 PRIORITY
013622  013711  001272                          CLR     DRVTYP          ;CLEAR DRIVE TYPE STORAGE
013626  005037  001240                          CLR     DRVSTA          ;SET DRIVE STATUS TO OFFLINE
013632  005037  001304                          CLR     DRVSTA+2        ;SET DRIVE STATUS TO OFFLINE
013636  005037  001306                          CLR     DRVSTA+4        ;SET DRIVE STATUS TO OFFLINE
013642  005037  001310                          CLR     DRVSTA+6        ;SET DRIVE STATUS TO OFFLINE
013646  005037  001312                          MOV     RPADR,R0        ;PUT RP11 ADDRESS INTO R0
013652  013700  001266                          MOV     #CLEAR,RPCS(R0)  ;CLEAR THE RP11
013656  012760  000001  000004                  MOV     #CLEAR,RPCS(R0)  ;CLEAR THE RP11 AGAIN
013664  012760  000001  000004                  MOV     #7,R1           ;'DRVSTA' TABLE INDEX
013672  012701  000007                  1$:     MOVB    R1,RPCS+1(R0)   ;SELECT DRIVE
013676  110160  000005                          BIT     #SUFU,RPDS(R0)  ;SEE IF DRIVE UNSAFE
013702  032760  001000  000000                  BEQ     2$              ;BR IF NOT UNSAFE
013710  001404                                  MOVB    #-1,DRVSTA(R1)  ;SET INDICATOR TO 'UNSAFE'
013712  112761  177777  001304                  BR      4$              ;EXIT
013720  000417                          2$:     BIT     #SUOL,RPDS(R0)  ;IS DRIVE ONLINE ?
013722  032760  040000  000000                  BEQ     5$              ;BR IF NOT
013730  001422                                  BIT     #SUSI,RPDS(R0)  ;SEEK INCOMPLETE SET ?
013732  032760  004000  000000                  BNE     3$              ;BR IF SET
013740  001004                                  BIT     #SURDY,RPDS(R0)  ;IS DRIVE READY ?
013742  032760  100000  000000                  BEQ     5$              ;BR IF NOT
013750  001412                          3$:     MOVB    #1,DRVSTA(R1)   ;SET DRIVE INDICATOR TO 'ONLINE'
013752  112761  000001  001304          4$:     BIT     #SURP03,RPDS(R0)  ;IS THE DRIVE AN RP03 ?
013760  032760  020000  000000                  BEQ     5$              ;BR IF NOT
013766  001403                                  BISB    ATABIT(R1),DRVTYP  ;SET RP03 INDICATOR
013770  156137  001314  001240          5$:     DEC     R1              ;DECREMENT THE TABLE INDEX
013776  005301                                  BPL     1$              ;BR IF NOT FINISHED
014000  100336                                  RESREG                  ;RESTORE R0-R5
014002  104413                                  RTS     PC              ;RETURN
014004  000207

                                        ;;*****************************************************************
                                        .SBTTL  SEEKS - INITIATE A DIRECT SEEK (OR HOME SEEK) OPERATION
                                        ;*THIS ROUTINE INITIATES A SEEK OPERATION ON THE DRIVE ADDRESSED IN
                                        ;*THE SECOND BYTE OF 'DPB'. 1 SECOND IS ALLOWED FOR THE SEEK
                                        ;*TO COMPLETE; IF THE DRIVE HAS NOT INTERRUPTED ('AIE' IS SET)
                                        ;*WITHIN THAT TIME, THE SUBSYSTEM IS RESET AND THE TESTING FOR THE
                                        ;*DRIVE IS TERMINATED.
                                        ;CALL
                                        ;       JSR     PC,@#SEEKS
                                        ;       RETURN

014006  104412                          SEEKS:  SAVREG                  ;SAVE R0-R5
014010  013737  001272  177776                  MOV     RPPRIO,PSW      ;RP11 PRIORITY
014016  013700  001266                          MOV     RPADR,R0        ;RP11 BUS ADDRESS
014022  013746  001110                          MOV     $LPERR,-(SP)    ;SAVE OLD LOOP ON ERROR ADDRESS
014026  012737  014034  001110                  MOV     #SEEKS0,$LPERR  ;CHANGE LOOP ON ERROR ADDRESS
014034  012760  000001  000004          SEEKS0: MOV     #CLEAR,RPCS(R0)  ;CLEAR THE CONTROLLER
014042  012760  000001  000004                  MOV     #CLEAR,RPCS(R0)  ;CLEAR THE CONTROLLER
014050  113760  001365  000005                  MOVB    DPB+1,RPCS+1(R0)  ;SELECT THE DRIVE
```

```
3614  014056  013737  001366  001262          MOV     DPB+2,CYL,DS    ;DESIRED CYLINDER
3615  014064  013760  001366  000012          MOV     DPB+2,RPCA(R0)  ;LOAD CYLINDER ADDRESS
3616  014072  013760  001370  000014          MOV     DPB+4,RPDA(R0)  ;TRACK ADDRESS
3617  014100  012777  014264  165162          MOV     #SEEKS1,@RPVEC  ;INTERRUPT RETURN
3618  014106  013760  001364  000004          MOV     DPB,RPCS(R0)    ;START THE COMMAND
3619  014114  152760  000040  000005          BISB    #40,RPCS+1(R0)  ;SET ATTENTION INTERRUPT ENABLE
3620  014122  005037  177776                  CLR     PS              ;ALLOW RP11 INTERRUPTS
3621  014126  010637  001242                  MOV     SP,SPSAV        ;SAVE THE CURRENT STACK POINTER
3622  014132  012737  004704  001300          MOV     #2500.,STALLG   ;SET STALL VALUE TO 2500MS
3623  014140  004037  015542                  JSR     R0,STALLA       ;WAIT THE SPECIFIED VALUE
3624  014144  001300                          .WORD   STALLG          ;STALL VALUE STORAGE
3625  014146  004737  015134                  JSR     PC,SAVRP        ;OPERATION TIMED OUT, SAVE THE REGISTERS
3626  014152  032737  001000  002004          BIT     #SUFU,$RPDS     ;DRIVE UNSAFE ?
3627  014160  001403                          BEQ     1$              ;BR IF NOT
3628  014162  104002                          ERROR   2               ;DRIVE UNSAFE
3629  014164  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
3630  014170  032737  040000  002004  1$:     BIT     #SUOL,$RPDS     ;DRIVE STILL ONLINE ?
3631  014176  001003                          BNE     2$              ;BR IF IT IS
3632  014200  104010                          ERROR   10              ;DRIVE OFFLINE AFTER POSITIONING
3633  014202  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
3634  014206  032737  100000  002004  2$:     BIT     #SURDY,$RPDS    ;IS THE DRIVE READY ?
3635  014214  001003                          BNE     3$              ;BR IF THE DRIVE IS READY
3636  014216  104016                          ERROR   16              ;DRIVE NOT READY AFTER POSITIONING
3637  014220  000137  015242                  JMP     DSELCT          ;DESELCT THE DRIVE
3638  014224  104005                  3$:     ERROR   5               ;NO INTERRUPT FROM POSITIONING AFTER 1 SECOND
3639  014226  032737  000200  002010          BIT     #RDY,$RPCS      ;IS THE CONTROLLER READY ?
3640  014234  001011                          BNE     4$              ;BR IF IT IS
3641  014236  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
3642  014244  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
3643  014252  012777  000100  164664          MOV     #BIT06,@STKS    ;RESTORE THE KEYBOARD INTERRUPT ENABLE
3644  014260  000177  164760          4$:     JMP     @BYPASS         ;TERMINATE THE PRESENT TEST
3645  014264  005037  177776          SEEKS1: CLR     PSW             ;GO BACK TO PRIORITY 0
3646  014270  013706  001242                  MOV     SPSAV,SP        ;RESTORE THE STACK POINTER
3647  014274  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
3648  014300  004737  015134                  JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
3649  014304  032737  004000  002004          BIT     #SUSI,$RPDS     ;SEEK INCOMPLETE ?
3650  014312  001416                          BEQ     SEEKS2          ;BR IF NOT
3651  014314  012737  014324  001210          MOV     #1$,$ESCAPE     ;ERROR 'ESCAPE' ADDRESS
3652  014322  104015                          ERROR   15              ;SEEK INCOMPLETE
3653  014324  005037  001210          1$:     CLR     $ESCAPE         ;CLEAR THE ESCAPE ADDRESS
3654  014330  004737  015264                  JSR     PC,RESTOR       ;DO A HOME SEEK TO CLEAR THE DRIVE
3655  014334  032777  001000  164576          BIT     #SWR9,@SWR      ;LOOPING ON ERROR ?
3656  014342  001452                          BEQ     SEEKS4          ;BR IF NOT
3657  014344  000177  164540                  JMP     @#LPERR         ;LOOP ON THE OPERATION
3658  014350  032737  100000  002010  SEEKS2: BIT     #ERR,$RPCS      ;'ERROR' SET ?
3659  014356  001410                          BEQ     1$              ;BR IF NOT
3660  014360  104006                          ERROR   6               ;DRIVE ERROR AFTER POSITIONING
3661  014362  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
3662  014370  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
3663  014376  000415                          BR      SEEKS3          ;CHECK THE DISK'S POSITION
3664  014400  005001                  1$:     CLR     R1              ;CLEAR R1
3665  014402  113701  001365                  MOVB    DPB+1,R1        ;DRIVE ADDRESS
3666  014406  136137  001314  002004          BITB    ATABIT(R1),$RPDS  ;ATTENTION BIT SET FOR THE DRIVE
3667  014414  001001                          BNE     2$              ;BR IF IT IS
3668  014416  104007                          ERROR   7               ;ATTENTION BIT NOT SET AFTER POSITIONING
3669  014420  023737  001366  002024  2$:     CMP     DPB+2,$SUCA     ;'SUCA' CORRECT ?
```

```
3670  014426  001401                          BEQ     SEEKS3          ;BR IF IT IS
3671  014430  104011                          ERROR   11              ;'SUCA' NOT CORRECT
3672  014432  004737  014524          SEEKS3: JSR     PC,VERIFY       ;VERIFY THE DISK'S POSITION
3673  014436  000414                          BR      SEEKS4          ;AT THE CORRECT CYLINDER
3674  014440  122737  000015  001364          CMPB    #HOMSEK,DPB     ;DOING A HOME SEEK ?
3675  014446  001410                          BEQ     SEEKS4          ;BR IF YES, NO SENSE DOING A HOME SEEK AGAIN
3676  014450  004737  015264                  JSR     PC,RESTOR       ;DO A HOME SEEK
3677  014454  032777  001000  164456          BIT     #SW09,@SWR      ;LOOP ON THE OPERATION ?
3678  014462  001402                          BEQ     SEEKS4          ;BR IF NOT
3679  014464  000177  164420                  JMP     @$LPERR         ;LOOP
3680  014470  032777  000010  164442  SEEKS4: BIT     #SW03,@SWR      ;STALL ?
3681  014476  001403                          BEQ     1$              ;BR IF NOT
3682  014500  004037  015546                  JSR     R0,STALL        ;STALL
3683  014504  001276                          .WORD   STALL1          ;2SMS STALL
3684  014506  016037  000024  001254  1$:     MOV     SUCA(R0),CYL.CR ;CURRENT CYLINDER ADDRESS
3685  014514  012637  001110                  MOV     (SP)+,$LPERR    ;RESTORE THE LOOP ON ERROR ADDRESS
3686  014520  104413                          RESREG                  ;RESTORE R0-R5
3687  014522  000207                          RTS     PC              ;RETURN
3688
3689                                  ;*****************************************************************
3690                                  .SBTTL  VERIFY - ROUTINE TO VERIFY THE POSITION AFTER A SEEK
3691                                  ;*THIS ROUTINE VERIFIES THE POSITION OF THE DRIVE AFTER A SEEK
3692                                  ;*BY READING A HEADER AT THE DESTINATION CYLINDER.  THE HEADER
3693                                  ;*READ IS FROM THE SECTOR WHICH IS 2 SECTORS FROM THE ADDRESS IN
3694                                  ;*THE 'SOT' COUNTER.  THE CYLINDER FIELD FROM 'DPB+2' IS COMPARED
3695                                  ;*WITH THE CYLINDER FIELD FROM THE HEADER.
3696                                  ;CALL
3697                                  ;       JSR     PC,VERIFY
3698                                  ;       RETURN                  ;NORMAL RETURN
3699                                  ;       ERROR RETURN            ;CYLINDER FIELDS DID NOT COMPARE
3700
3701  014524  104412          VERIFY: SAVREG                          ;SAVE R0-R5
3702  014526  013746  001110          MOV     $LPERR,-(SP)            ;SAVE THE LOOP ON ERROR ADDRESS
3703  014532  032777  000001  164400  BIT     #SW0,@SWR               ;CHECK POSITION ?
3704  014540  001171                  BNE     VERFY1                  ;BR IF NOT
3705  014542  012737  014550  001110  MOV     #VERFY0,$LPERR          ;LOOP ON ERROR ADDRESS
3706  014550  012737  177777  027142  VERFY0: MOV     #-1,BUFFER+2    ;CLEAR CYLINDER STORAGE
3707  014556  013700  001266          MOV     RPADR,R0                ;LOAD RP11 ADDRESS
3708  014562  016046  000014          MOV     RPDA(R0),-(SP)          ;GET THE 'SOT'
3709  014566  042716  177417          BIC     #^C360,(SP)             ;SAVE THE 'SOT' BITS
3710  014572  006216                  ASR     (SP)                    ;RIGHT JUSTIFY THE BITS
3711  014574  006216                  ASR     (SP)                    ;RIGHT JUSTIFY THE BITS
3712  014576  006216                  ASR     (SP)                    ;RIGHT JUSTIFY THE BITS
3713  014600  006216                  ASR     (SP)                    ;RIGHT JUSTIFY THE BITS
3714  014602  062716  000002          ADD     #2,(SP)                 ;FORM SECTOR ADDRESS FOR CHECK
3715  014606  022716  000012          CMP     #10.,(SP)               ;SEE IF THE ADDRESS OVERFLOWED
3716  014612  101005                  BHI     2$                      ;BR IF NOT
3717  014614  103402                  BLO     1$                      ;BR IF ADDR IS 11.
3718  014616  005016                  CLR     (SP)                    ;SET SECTOR ADDRESS TO ZERO
3719  014620  000402                  BR      2$                      ;CONTINUE
3720  014622  012716  000001  1$:     MOV     #1,(SP)                 ;SET SECTOR ADDRESS TO ONE
3721  014626  012777  014776  164434  2$:     MOV     #4$,@RPVEC      ;INTERRUPT ADDRESS
3722  014634  112637  001370          MOVB    (SP)+,DPB+4             ;PUT VALUE INTO THE DPB
3723  014640  113737  001370  001260  MOVB    DPB+4,SEC.RD            ;SELECTED SECTOR FOR ERROR MESSAGE
3724  014646  113737  001371  001256  MOVB    DPB+5,TRK.RD            ;TRACK FOR ERROR MESSAGE
3725  014654  013746  001364          MOV     DPB,-(SP)               ;SETUP RPCS LOAD VALUE
```

```
3726  014660  112716  000017                    MOVB    #READ,(SP)        ;READ COMMAND CODE
3727  014664  052716  014100                    BIS     #MODE!HDR!IDE,(SP)  ;SET BITS FOR HEADER READ
3728  014670  012760  177775  000006            MOV     #-3,RPWC(R0)      ;LOAD WORD COUNT
3729  014676  012760  027140  000010            MOV     #BUFFER,RPBA(R0)  ;BUFFER ADDRESS
3730  014704  013760  001370  000014            MOV     DPB+4,RPDA(R0)    ;SECTOR ADDRESS
3731  014712  012660  000004                    MOV     (SP)+,RPCS(R0)    ;START THE ORDER
3732  014716  005037  177776                    CLR     PSW               ;SET PRIORITY BACK TO ZERO
3733  014722  010637  001242                    MOV     SP,SPSAV          ;SAVE THE CURRENT STACK POINTER
3734  014726  012737  004704  001300            MOV     #2500.,STALLG     ;SET STALL VALUE TO 2500MS
3735  014734  004037  015542                    JSR     R0,STALLA         ;WAIT THE SPECIFIED VALUE
3736  014740  001300                            .WORD   STALLG            ;STALL VALUE STORAGE
3737  014742  004737  015134                    JSR     PC,SAVRP          ;STORE THE RP11 REGISTERS
3738  014746  104014                            ERROR   14                ;NO INTERRUPT WITHIN 1 SECOND
3739  014750  012760  000001  000004            MOV     #CLEAR,RPCS(R0)   ;CLEAR THE CONTROLLER
3740  014756  012760  000001  000004            MOV     #CLEAR,RPCS(R0)   ;CLEAR THE CONTROLLER
3741  014764  012777  000100  164152            MOV     #BIT6,@$TKS       ;SET TTY KEYBOARD INTERRUPT ENABLE
3742  014772  000177  164246                    JMP     @BYPASS           ;TERMINATE THE PRESENT TEST
3743  014776  005037  177776          4$:       CLR     PSW               ;SET PRIORITY TO ZERO
3744  015002  013706  001242                    MOV     SPSAV,SP          ;RESTORE STACK POINTER
3745  015006  004737  015134                    JSR     PC,SAVRP          ;STORE THE RP11 REGISTERS
3746  015012  013700  001266                    MOV     RPADR,R0          ;PUT RP11 ADDRESS INTO R0
3747  015016  032737  100000  002010            BIT     #ERR,$RPCS        ;DID THE ORDER TERMINTE WITH AN ERROR ?
3748  015024  001410                            BEQ     6$                ;BR IF NOT
3749  015026  104012                    5$:     ERROR   12                ;ERROR DURING POSITIONING VERIFICATION
3750  015030  012760  000001  000004            MOV     #CLEAR,RPCS(R0)   ;CLEAR THE ERROR
3751  015036  012760  000001  000004            MOV     #CLEAR,RPCS(R0)   ;CLEAR THE CONTROLLER
3752  015044  000427                            BR      VERFY1            ;BYPASS POSITIONING CHECK
3753  015046                            6$:
3754  015046  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3755  015052  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3756  015056  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3757  015062  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3758  015066  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3759  015072  006237  027142                    ASR     BUFFER+2          ;ALIGN CYLINDER ADDRESS FOR COMPARE
3760  015076  023737  001366  027142            CMP     DPB+2,BUFFER+2    ;SEE IF CYLINDER ADDR FROM HEADER IS CORRECT
3761  015104  001407                            BEQ     VERFY1            ;BR IF OK
3762  015106  013737  027142  001126            MOV     BUFFER+2,$BDDAT   ;CYLINDER ADDRESS FROM HEADER
3763  015114  104013                            ERROR   13                ;CYLINDER ADDRESS DOES NOT COMPARE
3764  015116  062766  000002  000002            ADD     #2,2(SP)          ;RETURN+2
3765  015124  012637  001110          VERFY1:   MOV     (SP)+,$LPERR      ;RESTORE LOOP ON ERROR ADDRESS
3766  015130  104413                            RESREG                    ;RESTORE R0-R5
3767  015132  000207                            RTS     PC                ;RETURN
3768
3769                                      ;;*****************************************************************
3770                                      ;SBTTL  SAVRP - STORE THE RP11E REGISTERS
3771                                      ;*THIS ROUTINE STORES THE RP11E REGISTERS FOR USE BY THE PROGRAM.
3772                                      ;*THE PROGRAM DOES NOT USE THE 'LIVE' REGISTERS FOR ERROR DETERMINATION.
3773                                      ;CALL
3774                                      ;       JSR     PC,SAVRP
3775                                      ;       RETURN
3776
3777  015134  010046                    SAVRP:  MOV     R0,-(SP)          ;SAVE R0
3778  015136  013700  001266                    MOV     RPADR,R0          ;RP11 ADDRESS
3779  015142  016037  000000  002004            MOV     RPDS(R0),$RPDS    ;STORE RPDS
3780  015150  016037  000002  002006            MOV     RPER(R0),$RPER    ;STORE RPER
3781  015156  016037  000004  002010            MOV     RPCS(R0),$RPCS    ;STORE RPCS
```

```
 3782  015164  016037  000006  002012          MOV     RPWC(R0),$RPWC  ;STORE RPWC
 3783  015172  016037  000010  002014          MOV     RPBA(R0),$RPBA  ;STORE RPBA
 3784  015200  016037  000012  002016          MOV     RPCA(R0),$RPCA  ;STORE RPCA
 3785  015206  016037  000014  002020          MOV     RPDA(R0),$RPDA  ;STORE RPDA
 3786  015214  016037  000016  002022          MOV     RPM1(R0),$RPM1  ;STORE RPM1
 3787  015222  016037  000024  002024          MOV     SUCA(R0),$SUCA  ;STORE SUCA
 3788  015230  016037  000026  002026          MOV     SILO(R0),$SILO  ;STORE SILO
 3789  015236  012600                          MOV     (SP)+,R0        ;RESTORE R0
 3790  015240  000207                          RTS     PC              ;RETURN
 3791
 3792                                  ;;*****************************************************************
 3793                                  ;SBTTL  DSELCT - SETUP INDICATORS TO DEASSIGN AN UNSAFE/OFFLINE DRIVE
 3794                                  ;*THIS ROUTINE SETS A BIT IN LOCATION 'DSELCT' WHICH IS USED TO DESELCT
 3795                                  ;*AN UNSAFE/OFFLINE DRIVE WHICH WAS ASSIGNED BY THE OPERATOR FOR TESTING
 3796                                  ;*OR WHICH WENT OFFLINE OR UNSAFE DURING TESTING.
 3797                                  ;CALL
 3798                                  ;       JMP     @#DSELCT
 3799
 3800  015242  013701  001246          DSELCT: MOV     CHKDRV,R1       ;GET DRIVE NUMBER
 3801  015246  156137  001314  001252          BISB    ATABIT(R1),DRVBAD  ;USE AS INDEX TO LOAD ATA BIT FOR DRIVE
 3802                                                                  ;IN DESELECT LOCATION.
 3803  015254  005037  177776                  CLR     PSW             ;SET PRIORITY BACK TO ZERO
 3804  015260  000137  007464                  JMP     @#SEOP          ;FORCE END OF PASS FOR THE DRIVE
 3805
 3806                                  ;;*****************************************************************
 3807                                  ;SBTTL  RESTOR - HOME SEEK ROUTINE
 3808                                  ;*THIS ROUTINE PERFORMS A HOME SEEK ON THE SELECTED DRIVE.  THE ROUTINE
 3809                                  ;*IS CALLED AFTER THE PROGRAM DETERMINES THAT THE DRIVE IS POSITIONED
 3810                                  ;*INCORRECTLY AFTER A SEEK OR A UTILITY HOME SEEK OPERATION IS REQUIRED.
 3811                                  ;CALL
 3812                                  ;       JSR     PC,RESTOR
 3813                                  ;       RETURN
 3814
 3815  015264  104412                  RESTOR: SAVREG                  ;SAVE R0-R5
 3816  015266  013746  001110                  MOV     $LPERR,-(SP)    ;SAVE LOOP ON ERROR ADDRESS
 3817  015272  012737  015304  001110          MOV     #1$,$LPERR      ;LOOP ON ERROR ADDRESS
 3818  015300  013700  001266                  MOV     RPADR,R0        ;RP11 ADDRESS
 3819  015304  012760  000001  000004  1$:     MOV     #CLEAR,RPCS(R0) ;CLEAR THE RP11
 3820  015312  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
 3821  015320  012777  015500  163742          MOV     #RESTRI,@RPVEC  ;INTERRUPT RETURN
 3822  015326  013746  001364                  MOV     DPB,-(SP)       ;SETUP RPCS CONTENTS
 3823  015332  112716  000015                  MOVB    #HOMSEK,(SP)    ;COMMAND CODE
 3824  015336  052716  020000                  BIS     #AIE,(SP)       ;ATTENTION INTERRUPT ENABLE
 3825  015342  012660  000004                  MOV     (SP)+,RPCS(R0)  ;START THE HOME SEEK
 3826  015346  005037  177776                  CLR     PSW             ;SET PRIORITY TO ZERO
 3827  015352  010637  001242                  MOV     SP,SPSAV        ;SAVE THE STACK POINTER
 3828  015356  012737  004704  001300          MOV     #2500.,STALLG   ;SET STALL VALUE TO 2500MS
 3829  015364  004037  015542                  JSR     R0,STALLA       ;WAIT THE SPECIFIED VALUE
 3830  015370  001300                          .WORD   STALLG          ;STALL VALUE STORAGE
 3831  015372  004737  015134                  JSR     PC,SAVRP        ;GET THE REGISTERS
 3832  015376  032737  001000  002004          BIT     #SUFU,$RPDS     ;DRIVE UNSAFE ?
 3833  015404  001403                          BEQ     2$              ;BR IF NOT
 3834  015406  104002                          ERROR   2               ;DRIVE UNSAFE
 3835  015410  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
 3836  015414  032737  040000  002004  2$:     BIT     #SUOL,$RPDS     ;IS THE DRIVE STILL ONLINE ?
 3837  015422  001003                          BNE     3$              ;BR IF IT IS
```

```
015424  104010                          ERROR   10              ;DRIVE OFFLINE AFTER POSITIONING
015426  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
015432  032737  100000  002004  3$:     BIT     #SDRDY,$RPDS    ;IS THE DRIVE READY ?
015440  001003                          BNE     4$              ;BR IF IT IS
015442  104016                          ERROR   16              ;DRIVE NOT READY AFTER POSITIONING
015444  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
015450  104005                  4$:     ERROR   5               ;DRIVE HAS NOT INTERRUPTED WITHIN 1 SEC
015452  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
015460  012760  000001  000004          MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
015466  012777  000100  163450          MOV     #BIT6,@$TKS     ;SET INTERRUPT ENABLE FOR TTY KEYBOARD
015474  000177  163544                  JMP     @BYPASS         ;TERMINATE THE PRESENT TEST
015500  013706  001242          RESTR1: MOV     SPSAV,SP        ;RESTORE THE STACK POINTER
015504  005037  177776                  CLR     PSW             ;RESET THE PSW
015510  004737  015134                  JSR     PC,SAVRP        ;SAVE THE REGISTERS
015514  032737  004000  002004          BIT     #SUSI,$RPDS     ;SEEK INCOMPLETE ?
015522  001403                          BEQ     2$              ;BR IF NOT
015524  104015                          ERROR   15              ;SEEK INCOMPLETE
015526  000137  015242                  JMP     DSELCT          ;DESELECT THE DRIVE
015532  012637  001110          2$:     MOV     (SP)+,$LPERR    ;RESTORE THE LOOP ON ERROR ADDRESS
015536  104413                          RESREG                  ;RESTORE R0-R5
015540  000207                          RTS     PC              ;RETURN

                        ;;*****************************************************************
                        .SBTTL  STALL - USED TO STALL AFTER A DRIVE FUNCTION IS COMPLETE
                        ;*THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
                        ;*AMOUNT OF TIME IF SWITCH 02 IS NOT SET OR A RANDOM AMOUNT OF TIME IF
                        ;*SWITCH 02 IS SET.
                        ;*STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0-6.
                        ;CALL
                        ;       JSR     R0,@#STALL
                        ;       TIME POINTER            ;WHERE TO FIND THE STALL TIME
                        ;
                        ;       JSR     R0,@#STALLA     ;ENTER HERE TO BYPASS RANDOM STALL SELECTION
                        ;       TIME POINTER            ;WHERE TO FIND THE STALL TIME

015542  013046          STALLA: MOV     @(R0)+,-(SP)    ;PICKUP STALL TIME
015544  000413                  BR      STALLB          ;BYPASS RANDOM STALL CHECK
015546  013046          STALL:  MOV     @(R0)+,-(SP)    ;PICKUP STALL TIME
015550  032777  000004  163362          BIT     #SW2,@SWR       ;USE A RANDOM TIME?
015556  001406                          BEQ     STALLB          ;NO--BRANCH
015560  004737  013510                  JSR     PC,@#$RAND      ;YES--FORM RANDOM NUMBER
015564  013716  013610                  MOV     @#$LONUM,(SP)   ;AND USE IT FOR THE STALL TIME
015570  042716  177700                  BIC     #^C77,(SP)      ;BUT NEVER > 64 MILLISECONDS
015574  005046          STALLB: CLR     -(SP)           ;CLEAR TEMP. LOCATION
015576  162766  000001  000002  2$:     SUB     #1,2(SP)        ;MORE STALL REQUIRED?
015604  103407                          BLO     4$              ;NO--BRANCH
015606  012716  000620                  MOV     #400.,(SP)      ;STALL  TIME CONSTANT
015612  005700                  3$:     TST     R0              ;NOP TO KILL TIME
015614  005366  000000                  DEC     0(SP)           ;COUNT
015620  001374                          BNE     3$              ;LOOP IF MORE COUNTS NEEDED
015622  000765                          BR      2$
015624  022626                  4$:     CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
015626  000200                          RTS     R0              ;EXIT

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

```
                                        .SBTTL  SORT - ROUTINE TO SORT & PRINT SEEK TIMES
                                        ;*THIS ROUTINE DETERMINES THE NUMBER OF TIMES A PARTICULAR FORWARD OR REVERSE
                                        ;*SEEK TIME WAS OBTAINED.  THE SEEK TIMES ARE SORTED IN DECENDING SEQUENCE
                                        ;*AND THE NUMBER OF OCCURENCES OF A PARTICULAR SEEK TIME ARE TOTALED.  THE
                                        ;*THE ROUTINE WILL TYPEOUT ONLY THE LONGEST 5 SEEK TIMES.  FORWARD SEEK TIMES
                                        ;*ARE STORED IN A TABLE STARTING AT 'FRWSTR'; REVERSE SEEK TIMES ARE STORED
                                        ;*IN A TABLE STARTING AT 'RVSTR'.
                                        ;CALL
                                        ;       JSR     R0,SORT         ;SORT AND TYPE THE SEEK TIMES
                                        ;       NUMBER OF ITEMS         ;NUMBER OF ITEMS IN THE TABLE (405 MAX)

015630  012037  016076          SORT:   MOV     (R0)+,7$        ;GET THE ENTRY COUNT
015634  104412                          SAVREG                  ;SAVE R0-R5
015636  012705  177776                  MOV     #-2,R5          ;COUNT FOR FORWARD & REVERSE
015642  012737  024010  001162          MOV     #FRWSTR,$REG0   ;INITIALIZE POINTER TO FORWARD SEEK TIMES
015650  012737  024012  001164          MOV     #FRWSTR+2,$REG1  ;SECOND FORWARD SEEK TIME POINTER
015656  012737  023510  001166          MOV     #BUFR4,$REG2
015664  012737  023570  001170          MOV     #BUFR5,$REG3    ;INITIALIZE POINTER TO '# OF TIMES'
015672  013700  001162          1$:     MOV     $REG0,R0        ;TYPE 'SEEK TIME'
015676  013701  001164                  MOV     $REG1,R1
015702  013702  016076                  MOV     7$,R2           ;NUMBER OF ITEMS IN THE TABLE
015706  005003                          CLR     R3
015710  021011                  2$:     CMP     (R0),(R1)       ;SORT THE ITEMS & PUT THEM
015712  003404                          BLE     3$              ;IN DESCENDING ORDER
015714  011004                          MOV     (R0),R4         ;LARGER ITEMS AT TOP OF LIST,
015716  011110                          MOV     (R1),(R0)       ;SMALLER AT THE BOTTOM
015720  010411                          MOV     R4,(R1)
015722  005203                          INC     R3
015724  005720                  3$:     TST     (R0)+
015726  005721                          TST     (R1)+
015730  005302                          DEC     R2
015732  001366                          BNE     2$
015734  005703                          TST     R3              ;SORTED ALL ITEMS'
015736  001355                          BNE     1$              ;IF NOT LOOP BACK
015740  013700  001162                  MOV     $REG0,R0        ;PTR TO 'SEEK TIME'
015744  013701  001166                  MOV     $REG2,R1        ;SAVE 'SEEK TIME' HERE
015750  013702  001170                  MOV     $REG3,R2        ;SAVE '# OF TIMES' HERE
015754  010204                          MOV     R2,R4
015756  005024                          CLR     (R4)+           ;CLR OUT 5 WORDS OF
015760  005024                          CLR     (R4)+           ;'# OF TIMES'BUFR
015762  005024                          CLR     (R4)+
015764  005024                          CLR     (R4)+
015766  005024                          CLR     (R4)+
015770  012703  177773                  MOV     #-5,R3          ;SAVE ONLY 5 DIFFERENT 'SEEK TIMES'
015774  011011                          MOV     (R0),(R1)       ;FIND OUT THE '# OF TIMES'
015776  013703  016076                  MOV     7$,R3           ;EACH 'SEEK TIME' WAS
016002  022011                  4$:     CMP     (R0)+,(R1)
016004  001411                          BEQ     5$              ;OBTAINED
016006  005721                          TST     (R1)+
016010  016011  177776                  MOV     -2(R0),(R1)     ;SAVE 'SEEK TIME'
016014  005722                          TST     (R2)+
016016  012712  000001                  MOV     #1,(R2)         ;KEEP '# OF TIMES'
016022  005303                          DEC     R3
016024  001404                          BEQ     6$
016026  000765                          BR      4$
016030  005212                  5$:     INC     (R2)            ;INCREMENT '# OF TIMES'
```

```
                                                DEC     R3              ;ALL DONE?
                                                BNE     4$              ;IF NOT, GO BACK
                                        6$:     INC     R5              ;SORTED BOTH FORWARD & REVERSE ?
                                                BEQ     GOTYPE          ;IF YES, TYPE THE VALUES
                                                MOV     #RVSTR,$REG0    ;IF NOT, INITIALIZE POINTER TO REVERSE SEEK TIMES
                                                MOV     #RVSTR+2,$REG1  ;SECOND REVERSE SEEK TIME POINTER
                                                MOV     #BUFR6,$REG2    ;SAVE 'SEEK TIME'
                                                MOV     #BUFR7,$REG3    ;SAVE '# OF TIMES' HERE
                                                JMP     1$              ;GO BACK & DO SORTING FOR REVERSE SEEK TIMES

                                        7$:     .WORD   0               ;ITEMS IN TABLE COUNT HERE

                                        ;TYPEOUT THE HEADER FOR THE SEEK TIME RESULTS

                                        GOTYPE:
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>/(SEEK TIME IS IN MICRO SECONDS)/
                                        64$:
                                                TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <15><12><12>/     FORWARD        REVERSE/
                                        66$:
                                                TYPE    ,69$            ;;TYPE ASCIZ STRING
                                                BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <15><12>/  # OF    SEEK  # OF   SEEK/
                                        68$:
                                                TYPE    ,71$            ;;TYPE ASCIZ STRING
                                                BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  <15><12>/  SEEKS    TIME  SEEKS   TIME/<15><12>
                                        70$:

                                        ;TYPEOUT THE '# OF SEEKS' & 'SEEK TIME' OBTAINED FOR EACH OF THOSE SEEKS

                                        TYPTIM: CLR     R0
                                                CLR     R5
                                        1$:     TYPE    ,$CRLF
                                                MOV     BUFR5(R0),-(SP) ;GET '# OF SEEKS', IF NONE (0)
                                                BEQ     3$              ;SKIP TYPING (FRWRD SEEK)
                                                TYPDS                   ;GO TYPE OUT DECIMAL '# OF SEEKS'
                                                TYPE
                                                BLNKS2
                                                MOV     BUFR4(R0),-(SP) ;GET 'SEEK TIME' FOR EACH OF
                                                TYPDS                   ;OF THAT '# OF SEEKS'. 'GO
                                                                        ;TYPE OUT IN DECIMAL
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  /0/
                                        64$:
                                        2$:     MOV     BUFR7(R0),-(SP)  ;GET '# OF SEEKS', IF NONE - 0
                                                BEQ     4$              ;SKIP TYPING (REVRSE SEEK)
                                                TST     R5              ;'FORWARD' VALUE FOR THIS LINE ?
                                                BEQ     6$              ;BR IF YES
                                                TYPE    ,BLNKS8         ;TYPE FILLER BLANKS
                                                TYPE    ,BLNKS7
                                        6$:     TYPDS                   ;TYPE OUT IN DECIMAL
```

```
4006  016422  104401  002043                    TYPE    ,BLNKS2          ;2 BLANKS
4007  016426  016046  023650                    MOV     BUFR6(R0),-(SP)  ;GET 'SEEK TIME' & TYPE IT
4008  016432  104405                            TYPDS                    ;OUT IN DECIMAL
4009  016434  104401  016442                    TYPE    ,67$             ;;TYPE ASCIZ STRING
4010  016440  000401                            BR      66$              ;;GET OVER THE ASCIZ
4011                                  ;;67$:    .ASCIZ  /0/
4012  016444                        66$:
4013  016444  000406                            BR      5$
4014  016446  005726                3$:         TST     (SP)+            ;POP STACK
4015  016450  005205                            INC     R5
4016  016452  000751                            BR      2$
4017  016454  005726                4$:         TST     (SP)+            ;POP STACK
4018  016456  005705                            TST     R5
4019  016460  001004                            BNE     7$
4020  016462  005720                5$:         TST     (R0)+            ;INCREMENT PTR TO TABLES
4021  016464  020027  000012                    CMP     R0,#10.          ;ALL DONE?
4022  016470  001323                            BNE     1$               ;IF NOT GO BACK
4023  016472  104413                7$:         RESREG                   ;RESTORE R0-R5
4024  016474  000200                            RTS     R0               ;RETURN
4025
4026                                  ;;*****************************************************************
4027                                  .SBTTL  SEEK TIME MEASUREMENT ROUTINE
4028                                  ;*THIS ROUTINE MEASURES THE TIME REQUIRED TO SEEK TO THE SPECIFIED CYLINDER.
4029                                  ;*THE ROUTINE USES THE 'SOT' COUNTER IN THE CONTROLLER TO MEASURE THE SEEK
4030                                  ;*TIME.  THE SEEK TIME MEASUREMENT IS BASED ON THE NUMBER OF SECTORS
4031                                  ;*OCCURING BETWEEN SEEK START AND SEEK COMPLETE.  THE RESOLUTION OF THIS
4032                                  ;*MEASUREMENT METHOD IS + OR - 2.5 MS (ONE SECTOR TIME).
4033                                  ;CALL
4034                                  ;       JSR     PC,TIMSEK        ;TIME THE SEEK TO THE CYLINDER IN DPB+2
4035                                  ;       RETURN                   ;SEEK TIME IS RETURNED IN R3
4036
4037
4038  016476  010046                TIMSEK:   MOV     R0,-(SP)         ;SAVE R0
4039  016500  010246                          MOV     R2,-(SP)         ;SAVE R2
4040  016502  013746  001110                  MOV     $LPERR,-(SP)     ;SAVE CURRENT LOOP ON EPROR ADDRESS
4041  016506  012737  016520  001110          MOV     #TIMSK1,$LPERR   ;NEW LOOP ON ERROR ADDRESS
4042  016514  013700  001266                  MOV     RPADR,R0         ;RP11 BASE ADDRESS
4043  016520  005003                TIMSK1:   CLR     R3               ;R3 WILL BE USED TO COUNT THE DISK REVOLUTIONS
4044  016522  012737  000062  001300          MOV     #50.,STALLG      ;SET STALL VALUE TO 50MS
4045  016530  004037  015542                  JSR     R0,STALLA        ;WAIT THE SPECIFIED VALUE
4046  016534  001300                          .WORD   STALLG           ;STALL VALUE STORAGE
4047  016536  016002  000014          1$:     MOV     RPDA(R0),R2      ;READ THE SECTOR/TRACK REGISTER
4048  016542  032702  000200                  BIT     #BIT7,R2         ;WAIT FOR SECTOR 8, THEN START LOOKING
4049  016546  001773                          BEQ     1$               ;FOR SECTOR 0.
4050  016550  016002  000014          2$:     MOV     RPDA(R0),R2      ;READ THE SECTOR REGISTER
4051  016554  026002  000014                  CMP     RPDA(R0),R2      ;WAS IT CHANGING ?
4052  016560  001373                          BNE     2$               ;BR IF IT WAS
4053  016564  032702  000360                  BIT     #360,R2          ;WAIT FOR SEC 0, INDEX MARK
4054  016566  001370                          BNE     2$               ;AS SOON AS IT IS AT SECTOR 0, START
4055                                                                   ;A SEEK & BEGIN TIMING
4056  016570  012777  017036  162472          MOV     #TIMSK2,@RPVEC   ;INTERRUPT ADDRESS
4057  016576  012760  000377  000000          MOV     #377,RPDS(R0)    ;CLEAR ANY ATTENTION BITS
4058  016604  013760  001366  000012          MOV     DPB+2,RPCA(R0)   ;LOAD CYLINDER ADDRESS
4059  016612  013746  001364                  MOV     DPB,-(SP)        ;SETUP COMMAND
4060  016616  052716  020000                  BIS     #AIE,(SP)        ;SET ATTENTION INTERRUPT ENABLE
4061  016622  012660  000004                  MOV     (SP)+,RPCS(R0)   ;ISSUE A SEEK, START TIMING
```

```
4062  016626  005037  177776                CLR     PSW             ;SET PRIORITY TO ZERO
4063  016632  016002  000014          3$:   MOV     RPDA(R0),R2     ;READ THE SECTOR REGISTER
4064  016636  026002  000014                CMP     RPDA(R0),R2     ;WAS IT CHANGING ?
4065  016642  001373                       BNE     3$              ;BR IF IT WAS
4066  016644  032702  000360                BIT     #360,R2         ;WAIT FOR SEC CNTR TO MOVE
4067  016650  001770                       BEQ     3$              ;FROM 0 TO 1
4068  016652  016002  000014          4$:   MOV     RPDA(R0),R2     ;READ THE SECTOR REGISTER
4069  016656  026002  000014                CMP     RPDA(R0),R2     ;WAS IT CHANGING ?
4070  016662  001373                       BNE     4$              ;BR IF IT WAS
4071  016664  032702  000360                BIT     #360,R2         ;WAIT FOR SECTOR 0
4072  016670  001370                       BNE     4$              ;BR IF NOT AT 0
4073  016672  005203                       INC     R3              ;COUNT A DISK REVOLUTION (25MS)
4074  016674  022703  000050                CMP     #40.,R3         ;1 SECOND ELAPSED ?
4075  016700  003354                       BGT     3$              ;BR IF NOT
4076  016702  004737  015134                JSR     PC,SAVRP        ;SAVE THE REGISTERS
4077  016706  032737  001000  002004        BIT     #SUFU,$RPDS     ;FILE UNSAFE ?
4078  016714  001403                       BEQ     5$              ;BR IF NOT
4079  016716  104002                       ERROR   2               ;REPORT DRIVE UNSAFE
4080  016720  000137  015242                JMP     DSELCT          ;SETUP TO DESELECT THE DRIVE
4081  016724                          5$:
4082  016724  013737  002004  001126        MOV     $RPDS,$BDDAT    ;GET CONTENTS OF RPDS
4083  016732  012737  000000  001122        MOV     #RPDS,$BDADR  ;FORM REGISTER ADDRESS FOR ERROR MESSAGE
4084  016740  063737  001266  001122        ADD     RPADR,$BDADR    ;ADD RP11 BASE ADDRESS
4085  016746  012737  140000  001124        MOV     #SURDY!SUOL,$GDDAT  ;WHAT REGISTER SHOULD BE
4086  016754  013737  001126  001204        MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
4087  016762  042737  037777  001204        BIC     #^C140000,$TMP0 ;SAVE THE SPECIFIED BITS
4088  016770  023737  001124  001204        CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
4089  016776  001403                       BEQ     6$              ;BR IF OK
4090  017000  104017                       ERROR   17              ;NOT READY/OFFLINE DURING TIMING TEST
4091  017002  000137  015242                JMP     DSELCT          ;DESELECT THE DRIVE
4092  017006  104005                  6$:  ERROR   5               ;NO INTERRUPT AFTER 1 SECOND
4093  017010  012760  000001  000004        MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
4094  017016  012760  000001  000004        MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
4095  017024  012777  000100  162112        MOV     #BIT06,@$TKS    ;SET KEYBOARD INTERRUPT ENABLE
4096  017032  000177  162206                JMP     @BYPASS         ;TERMINATE THE TEST
4097  017036  012626                  TIMSK2: MOV     (SP)+,(SP)+     ;RESTORE THE STACK POINTER
4098  017040  016002  000014                MOV     RPDA(R0),R2     ;SAVE RPDA
4099  017044  005037  177776                CLR     PSW             ;SET PRIORITY TO ZERO
4100  017050  032702  000360                BIT     #360,R2         ;AT SECTOR 0 ?
4101  017054  001001                       BNE     1$              ;BR IF NOT
4102  017056  005203                       INC     R3              ;INCREMENT THE REVOLUTION COUNTER
4103  017060  004737  015134          1$:   JSR     PC,SAVRP        ;STORE THE REGISTERS
4104  017064  032737  004000  002004        BIT     #SUSI,$RPDS     ;SEEK INCOMPLETE ?
4105  017072  001405                       BEQ     2$              ;BR IF NOT
4106  017074  104015                       ERROR   15              ;SEEK INCOMPLETE
4107  017076  004737  015264                JSR     PC,RESTOR       ;DO A HOME SEEK
4108  017102  000177  162136                JMP     @BYPASS         ;TERMINATE THE PRESENT TEST
4109  017106  032737  100000  000004  2$:   BIT     #ERR,RPCS       ;OTHER ERRORS ?
4110  017114  001412                       BEQ     SKDON           ;BR IF NOT
4111  017116  104004                       ERROR   4               ;REPORT ERROR DURING TIMING TEST
4112  017120  012760  000001  000004        MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
4113  017126  012760  000001  000004        MOV     #CLEAR,RPCS(R0) ;CLEAR THE CONTROLLER
4114  017134  113760  001365  000005        MOVB    DPB+1,RPCS+1(R0)  ;RESELECT THE DRIVE
4115  017142  042702  177417          SKDON: BIC    #^C360,R2       ;CLEAR SECTOR/TRACK BITS - LEAVE THE 'SOT'
4116  017146  006202                       ASR     R2              ;RIGHT JUSTIFY 'SOT'
4117  017150  006202                       ASR     R2              ;RIGHT JUSTIFY 'SOT'
```

```
 4118  017152  006202                        ASR     R2              ;RIGHT JUSTIFY 'SOT'
 4119  017154  006202                        ASR     R2              ;RIGHT JUSTIFY 'SOT'
 4120  017156  012746  000010                MOV     #10.,-(SP)              ;;PUT THE MULTIPLER ON THE STACK
 4121  017162  010346                        MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
 4122  017164  004737  013022                JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
 4123  017170  012616                        MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
 4124  017172  012603                        MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
 4125  017174  060203                        ADD     R2,R3           ;ADD PARTIAL SECTOR COUNT
 4126                                                                ;NOTE THERE IS A SCALE FACTOR
 4127  017176  012746  000372                MOV     #250.,-(SP)             ;;PUT THE MULTIPLER ON THE STACK
 4128  017202  010346                        MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
 4129  017204  004737  013022                JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
 4130  017210  012616                        MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
 4131  017212  012603                        MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
 4132
 4133                          ;;************************************************************
 4134                          ;SEEK TIME (IN US) = [(R3)X(10)+(R2)]X(250)X(10)
 4135                          ;;************************************************************
 4136
 4137  017214  012637  001110                MOV     (SP)+,$LPERR    ;RESTORE THE LOOP ON ERROR ADDRESS
 4138  017220  012602                        MOV     (SP)+,R2        ;RESTORE R2
 4139  017222  012600                        MOV     (SP)+,R0        ;RESTORE R0
 4140  017224  000207                        RTS     PC              ;RETURN
 4141
 4142                          ;;************************************************************
 4143                          .SBTTL  GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
 4144                          ;*THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
 4145                          ;*OF THE RP11 IS SETUP TO READ THE PROPER VALUE.
 4146                          ;*IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 4147                          ;*REQUIRED.
 4148                          ;NOTE: THIS ROUTINE DESTROYS R0-R4
 4149                          ;CALL
 4150                          ;
 4151                          ;       JSR     PC,@#GETADR
 4152                          ;       RETURN
 4153
 4154  017226  005737  001224  GETADR: TST     @#BUSADR        ;INPUT FROM TTY REQUESTED?
 4155  017232  001447                        BEQ     7$              ;NO--BRANCH
 4156  017234  005037  001224                CLR     @#BUSADR        ;YES--CLEAR THE REQUEST FLAG
 4157  017240  012700  001266  1$:           MOV     #RPADR,R0       ;FIRST ADDRESS
 4158  017244  012703  021410                MOV     #MRPCS,R3       ;"RPADR="
 4159  017250  011004                        MOV     (R0),R4         ;PRESENT RP11 ADDRESS
 4160  017252  004037  017430                JSR     R0,@#GETNUM     ;GET NEW RP11 ADDRESS
 4161  017256  000402                        BR      2$              ;COMMA
 4162  017260  000767                        BR      1$              ;PERIOD
 4163  017262  000430                        BR      5$              ;DOUBLE PERIOD
 4164  017264  010420  2$:                   MOV     R4,(R0)+        ;SAVE NEW RHCS1
 4165  017266  012703  021420                MOV     #MRPVEC,R3      ;"RPVEC="
 4166  017272  011004                        MOV     (R0),R4         ;PRESENT RP11 VECTOR ADDRESS
 4167  017274  004037  017430                JSR     R0,@#GETNUM     ;GET NEW RPVEC
 4168  017300  000402                        BR      3$              ;COMMA
 4169  017302  000756                        BR      1$              ;PERIOD
 4170  017304  000417                        BR      5$              ;DOUBLE PERIOD
 4171  017306  010420  3$:                   MOV     R4,(R0)+        ;SAVE NEW RPVEC
 4172  017310  012703  021431                MOV     #MRPPRI,R3      ;"RPPRIO="
 4173  017314  011004                        MOV     (R0),R4         ;PRESENT RP11 PRIORITY LEVEL
```

```
 4174  017316  006304                          ASL     R4              ;POSITION FOR TYPEOUT
 4175  017320  006304                          ASL     R4
 4176  017322  006304                          ASL     R4
 4177  017324  000304                          SWAB    R4
 4178  017326  004037  017430                  JSR     R0,@#GETNUM     ;GET NEW RPPRIO
 4179  017332  000402                          BR      4$              ;COMMA
 4180  017334  000401                          BR      4$              ;PERIOD
 4181  017336  000404                          BR      6$              ;DOUBLE PERIOD
 4182  017340  010210                  4$:     MOV     R2,(R0)         ;SAVE NEW RPPRIO
 4183  017342  000736                          BR      1$              ;LOOP
 4184  017344  010410                  5$:     MOV     R4,(R0)         ;SAVE INPUT
 4185  017346  000401                          BR      7$
 4186  017350  010210                  6$:     MOV     R2,(R0)         ;SAVE PRIORITY
 4187  017352  013701  000004          7$:     MOV     @#ERRVEC,R1     ;SAVE THE ERROR VECTOR
 4188  017356  012737  017376  000004          MOV     #8$,@#ERRVEC    ;SETUP FOR TRAP
 4189  017364  005777  161676                  TST     @RPADR          ;CHECK FOR RP11
 4190  017370  010137  000004                  MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 4191  017374  000207                          RTS     PC              ;RETURN
 4192  017376  010137  000004          8$:     MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 4193  017402  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
 4194  017404  104001                          ERROR   1               ;REPORT THE ERROR
 4195  017406  005737  000042                  TST     @#42            ;IS THERE A MONITOR?
 4196  017412  001712                          BEQ     1$              ;NO--GO ASK FOR ADDRESS
 4197  017414  005037  001232                  CLR     @#DRVSEL        ;YES--NO DRIVES SELECTED
 4198  017420  005037  007640                  CLR     @#SEOPCT        ;NO PASSES
 4199  017424  000137  007464                  JMP     @#SEOP          ;GO TO END OF PROGRAM
 4200
 4201                                  ;;*****************************************************************
 4202                                  ;SBTTL  GETNUM - ROUTINE TO GET A NUMBER
 4203                                  ;*THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
 4204                                  ;*INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
 4205                                  ;*IF REQUIRED.
 4206                                  ;*NOTE: THIS ROUTINE DESTROYS R1
 4207                                  ;CALL
 4208                                  ;       MOV     #ADR,R3         ;ADDRESS OF ASCIZ MESSAGE
 4209                                  ;       MOV     #NUM,R4         ;OCTAL NUMBER
 4210                                  ;       JSR     R0,@#GETNUM
 4211                                  ;       RETURN1                 ;INPUT TERMINATED WITH A COMMA
 4212                                  ;       RETURN2                 ;WITH A PERIOD
 4213                                  ;       RETURN3                 ;WITH A DOUBLE PERIOD
 4214                                  ;                               ;R4=INPUT NUMBER AND
 4215                                  ;                               ;R2=R4*32 FOR ALL
 4216                                  ;                               ;THREE RETURNS
 4217
 4218  017430  010337  017436  GETNUM: MOV     R3,2$           ;SAVE MESSAGE POINTER
 4219  017434  104401          1$:     TYPE                    ;TYPE THE MESSAGE
 4220  017436  000000          2$:     .WORD   0               ;MESSAGE POINTER GOES HERE
 4221  017440  010446                  MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
 4222  017442  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4223  017444  104411                  RDLIN                   ;READ AN ASCIZ STRING
 4224  017446  012601                  MOV     (SP)+,R1        ;ADDRESS OF FIRST CHARACTER
 4225  017450  004037  021014          JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
 4226  017454  017434                  1$                      ;ILLEGAL CHARACTER
 4227  017456  017434                  1$                      ;CARRIAGE RETURN
 4228  017460  017470                  3$                      ;"/"
 4229  017462  017514                  7$                      ;","
```

```
4230  017464  017522                        8$                      ;"."
4231  017466  017434                        1$                      ;DIGIT 0-9
4232  017470                          3$:
4233  017470  004037  021250                JSR     R0,@#CK.NUM     ;CHECK THE NUMBER
4234  017474  017434                        1$                      ;ILLEGAL INPUT
4235  017476  017510                        6$                      ;TERMINATED WITH A ","
4236  017500  017506                        5$                      ;TERMINATED WITH A ":"
4237  017502  017504                        4$                      ;TERMINATED WITH A ".."
4238  017504  005720                  4$:   TST     (R0)+             ;DOUBLE PERIOD
4239  017506  005720                  5$:   TST     (R0)+             ;SINGLE PERIOD
4240  017510  010204                  6$:   MOV     R2,R4             ;COMMA--SAVE INPUT NUMBER
4241  017512  000414                        BR      10$               ;GO TO EXIT
4242  017514  105711                  7$:   TSTB    (R1)              ;TERMINATOR AFTER A COMMA?
4243  017516  001346                        BNE     1$                ;NO--LOOP
4244  017520  000411                        BR      10$               ;YES--EXIT
4245  017522  105711                  8$:   TSTB    (R1)              ;TERMINATOR AFTER A PERIOD?
4246  017524  001406                        BEQ     9$                ;YES--EXIT
4247  017526  122721  000056                CMPB    #'.,(R1)+         ;NO--DOUBLE PERIOD?
4248  017532  001340                        BNE     1$                ;NO--LOOP
4249  017534  105711                        TSTB    (R1)              ;YES--TERMINATOR?
4250  017536  001336                        BNE     1$                ;NO--LOOP
4251  017540  005720                        TST     (R0)+             ;DOUBLE PERIOD
4252  017542  005720                  9$:   TST     (R0)+             ;PERIOD
4253  017544  010402                  10$:  MOV     R4,R2             ;COMMA--POSITION THE
4254  017546  000302                        SWAB    R2                ;NUMBER IN CASE IT
4255  017550  006202                        ASR     R2                ;IS THE PRIORITY LEVEL
4256  017552  006202                        ASR     R2
4257  017554  006202                        ASR     R2
4258  017556  000200                        RTS     R0                ;EXIT
4259
4260                                  ;;*****************************************************************
4261                                  .SBTTL  GT.PRM - GET PARAMETERS
4262                                  ;*THIS ROUTINE IS USED TO CHANGE OR MODIFY
4263                                  ;*THE TEST PARAMETERS.  IT GIVES THE OPERATOR
4264                                  ;*THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
4265                                  ;*TESTS TO RUN AND HOW MANY TIMES TO
4266                                  ;*REPEAT EACH TEST
4267
4268  017560  104412                  GT.PRM: SAVREG                  ;SAVE R0-R5
4269  017562  005037  001232          GT.PR1: CLR     DRVSEL          ;NO DRIVE SELECTED
4270  017566  104401  017574                  TYPE    ,65$            ;;TYPE ASCIZ STRING
4271  017572  000406                        BR      64$               ;;GET OVER THE ASCIZ
4272
4273  017610                          ;;65$:  .ASCIZ  <15><12>/DRIVE(S)=/
4274  017610                          64$:
4275  017610  104411                        RDLIN                   ;READ TTY
4276  017612  012601                        MOV     (SP)+,R1        ;ADDRESS OF ASCIZ STRING
4277  017614  004037  021014                JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
4278  017620  017562                        GT.PR1                  ;ILLEGAL CHARACTER
4279  017622  017562                        GT.PR1                  ;CARRIAGE RETURN
4280  017624  017562                        GT.PR1                  ;"/"
4281  017626  017562                        GT.PR1                  ;","
4282  017630  017562                        GT.PR1                  ;"-"
4283  017632  017634                        1$                      ;DIGIT 0-9
4284  017634  005301                  1$:   DEC     R1
4285  017636                          2$:
4286  017636  012702  000007                MOV     #7,R2   ;UPPER LIMIT OF INPUT
```

```
4286  017642  004037  021070                    JSR     R0,@#CK.DIG     ;CHECK THE DIGIT(S)
4287  017646  017562                            GT.PR1                  ;ILLEGAL INPUT
4288  017650  017562                            GT.PR1                  ;INPUT TO LARGE
4289  017652  017660                            3$                      ;TERMINATED WITH A ","
4290  017654  017700                            4$                      ;TERMINATED WITH A ":"
4291  017656  017700                            4$                      ;TERMINATED WITH A "="
4292  017660  156237  001314  001232  3$:      BISB    ATABIT(R2),DRVSEL ;SET THE DRIVE SELECTED BIT
4293  017666  105741                            TSTB    -(R1)           ;WAS THE LINE TERMINATED?
4294  017670  001362                            BNE     2$              ;NO-GET THE NEXT DRIVE
4295  017672  005037  001234                    CLR     @#TSTNMS        ;DESELECT ALL TESTS
4296  017676  000405                            BR      GTTST1          ;YES--SELECT TEST
4297  017700  156237  001314  001232  4$:      BISB    ATABIT(R2),DRVSEL ;SET THE SELECTED DRIVE BITS
4298  017706  104413                  GT.PR2:   RESREG                  ;RESTORE R0-R5
4299  017710  000207                            RTS     PC              ;EXIT
4300
4301  017712                            GTTST1:
4302  017712  104401  017720                    TYPE    ,65$            ;;TYPE ASCIZ STRING
4303  017716  000403                            BR      64$             ;;GET OVER THE ASCIZ
4304                                    ;;65$:  .ASCIZ  /TEST=/
4305  017726                            64$:
4306  017726  104411                            ROLIN                   ;READ AN ASCIZ STRING
4307  017730  012601                            MOV     (SP)+,R1        ;POINTER TO R1
4308  017732  122711  000056                    CMPB    #'.,(R1)        ;DOUBLE PERIOD?
4309  017736  001007                            BNE     1$              ;NO--BRANCH
4310  017740  122761  000056  000001            CMPB    #'.,1(R1)
4311  017746  001003                            BNE     1$
4312  017750  105761  000002                    TSTB    2(R1)           ;"CR"?
4313  017754  001754                            BEQ     GT.PR2          ;YES--EXIT
4314  017756  005037  001234            1$:     CLR     TSTNMS          ;NO TEST SELECTED
4315  017762  005003                            CLR     R3              ;NO TEST TO BE OPENED
4316  017764  005004                  GTTST2:   CLR     R4              ;NO TEST BITS SET
4317  017766  004037  020740                    JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
4318  017772  000460                            BR      GTTST4          ;NO--BRANCH
4319  017774  010205                            MOV     R2,R5           ;YES--SAVE IT
4320  017776  005201                            INC     R1              ;MOVE TO NEXT CHARACTER
4321  020000  004037  020740                    JSR     R0,@#CK.OCT     ;OCTAL DIGIT
4322  020004  000405                            BR      1$              ;NO--BRANCH
4323  020006  005201                            INC     R1              ;MOVE TO NEXT CHARACTER
4324  020010  006305                            ASL     R5              ;SCALE HIGH DIGIT
4325  020012  006305                            ASL     R5
4326  020014  006305                            ASL     R5
4327  020016  060502                            ADD     R5,R2           ;COMBINE HIGH & LOW DIGITS
4328  020020  020227  000012            1$:     CMP     R2,#$TN-1       ;LEGAL TEST NUMBER?
4329  020024  003332                            BGT     GTTST1          ;NO--BRANCH
4330  020026  006302                            ASL     R2
4331  020030  016204  001324                    MOV     BITS(R2),R4     ;SELECT TEST
4332  020034  121127  000055                    CMPB    (R1),#'-        ;TEST STRING?
4333  020040  001035                            BNE     GTTST4          ;NO--BRANCH
4334  020042  005201                            INC     R1              ;YES--MOVE TO NEXT CHARACTER
4335  020044  004037  020740                    JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
4336  020050  000720                            BR      GTTST1          ;NO--BRANCH
4337  020052  010205                            MOV     R2,R5           ;YES--SAVE IT
4338  020054  005201                            INC     R1              ;MOVE TO NEXT CHARACTER
4339  020056  004037  020740                    JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
4340  020062  000405                            BR      2$              ;NO--BRANCH
4341  020064  005201                            INC     R1              ;YES--MOVE TO NEXT CHARACTER
```

```
4342  020066  006305                          ASL     R5              ;SCALE HIGH DIGIT
4343  020070  006305                          ASL     R5
4344  020072  006305                          ASL     R5
4345  020074  060502                          ADD     R5,R2           ;COMBINE HIGH & LOW DIGIT
4346  020076  020227  000012          2$:     CMP     R2,#STN-1       ;LEGAL TEST NUMBER?
4347  020102  003303                          BGT     GTTST1          ;NO--BRANCH
4348  020104  006302                          ASL     R2
4349  020106  020462  001324                  CMP     R4,BITS(R2)     ;IS THE FIRST NUMBER OF THE
```

```
4350                                                                    ;STRING SMALLER THAN THE LAST?
4351  020112  002277                          BGE     GTTST1            ;NO--BRANCH
4352  020114  056204  001324        3$:       BIS     BITS(R2),R4       ;YES--SET TEST SELECT BIT FOR SELECTED TESTS
4353  020120  005742                          TST     -(R2)             ;ADJUST POINTER
4354  020122  036204  001324                  BIT     BITS(R2),R4       ;DONE?
4355  020126  001772                          BEQ     3$                ;NO--BRANCH
4356  020130  000401                          BR      GTTST4            ;SKIP ... INCREMENT
4357  020132  005201                GTTST3:   INC     R1                ;MOVE TO NEXT CHARACTER
4358  020134  050437  001234        GTTST4:   BIS     R4,TSTNMS         ;SET SELECTED TEST BITS INTO
4359                                                                    ;TEST SELECT WORD
4360  020140  121127  000056                  CMPB    (R1),#'.          ;"PERIOD"?
4361  020144  001420                          BEQ     GTTST5            ;YES--BRANCH
4362  020146  005704                          TST     R4                ;ANY TEST SELECTED THIS CYCLE?
4363  020150  001660                          BEQ     GTTST1            ;NO--BRANCH
4364  020152  121127  000057                  CMPB    (R1),#'/          ;"OPEN"?
4365  020156  001002                          BNE     1$                ;NO--BRANCH
4366  020160  050403                          BIS     R4,R3             ;YES--SET BITS FOR TEST TO OPEN
```

```
 4367  020162  000403                          BR      2$
 4368  020164  121127  000054          1$:     CMPB    (R1),#',        ;"COMMA"?
 4369  020170  001250                          BNE     GTTST1          ;NO--BRANCH
                                       2$:     INC     R1              ;MOVE TO NEXT CHARACTER
                                               TSTB    (R1)            ;"CR"?
                                               BNE     GTTST2          ;NO--GO GET NEXT CHARACTER
                                               TST     R3              ;ANY TESTS TO OPEN?
                                               BNE     OPNTST          ;YES--BRANCH
                                               BR      GTTST1          ;NO--START AGAIN
                                       GTTST5: INC     R1              ;MOVE TO NEXT CHARACTER
 4377  020210  121127  000056                  CMPB    (R1),#'.        ;"PERIOD"?
                                               BEQ     GTTST6          ;YES--BRANCH
                                               TSTB    (R1)            ;"CR"?
                                               BEQ     1$              ;YES--BRANCH
                                               JMP     GTTST1
                                       1$:     TST     R3              ;ANY TEST TO BE OPENED?
                                               BNE     OPNTST          ;YES--BRANCH
                                               JMP     GT.PR2          ;NO--GO START TESTING
                                       GTTST6: INC     R1              ;MOVE TO NEXT CHARACTER
                                               TSTB    (R1)            ;"CR"?
                                               BEQ     1$              ;YES--BRANCH
                                               JMP     GTTST1          ;NO--GO ASK FOR TEST
                                       1$:     TST     R3              ;ANY TEST TO BE OPENED?
                                               BNE     OPNTST          ;YES--BRANCH
                                               JMP     GT.PR2          ;NO--GO START TESTING
                                       ;OPEN THE TEST FOR CHANGES
                                       OPNTST: SAVREG                  ;SAVE R0-R5
                                               CLR     (PC)+           ;START WITH TEST 0
                                       OPN.CT: .WORD   0               ;COUNT STORED HERE
                                               BR      OPN.2           ;SKIP THE INCREMENT
 4397  020270  005237  020264          OPN.1:  INC     OPN.CT          ;MOVE TO THE NEXT TEST
 4398  020274  022737  000012  020264          CMP     #STN-1,OPN.CT   ;TEST NUMBER TO BIG?
 4399  020302  002003                          BGE     OPN.2           ;NO--OPEN THE NEXT TEST
 4400  020304  104413                          RESREG                  ;RESTORE R0-R5
 4401  020306  000137  017712                  JMP     GTTST1          ;YES--GO ASK FOR MORE TESTS
 4402  020312  013705  020264          OPN.2:  MOV     OPN.CT,R5       ;SETUP TO USE THE
 4403  020316  006305                          ASL     R5              ;TEST NUMBER AS AN INDEX
 4404  020320  036503  001324                  BIT     BITS(R5),R3     ;OPEN THIS TEST?
 4405  020324  001761                          BEQ     OPN.1           ;NO--MOVE TO NEXT TEST
 4406  020326  104401  020334                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 4407  020332  000404                          BR      64$             ;;GET OVER THE ASCIZ
 4408                                 ;;65$:  .ASCIZ  / TEST /
 4409  020344                         64$:
 4410  020344  013746  020264                  MOV     OPN.CT,-(SP)    ;;SAVE OPN.CT FOR TYPEOUT
 4411                                                                  ;;TEST NUMBER
 4412  020350  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 4413  020352     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
 4414  020353     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 4415  020354  104401  001217                  TYPE    ,$CRLF          ;TYPE "CR" & "LF"
 4416  020360  016500  001406                  MOV     PRMPT(R5),R0    ;PICKUP PARAMETER POINTER
 4417  020364  011046                          MOV     (R0),-(SP)      ;SAVE THE VARIABLE INDICATOR
 4418  020366  012702  001372                  MOV     #PRM,R2         ;FIRST ADDRESS OF TABLE
 4419  020372  000405                          BR      2$
 4420  020374  006216                  1$:     ASR     (SP)            ;CHECK FOR A VARIABLE
 4421  020376  103403                          BCS     2$              ;GO MOVE THIS ONE
 4422  020400  001404                          BEQ     OPNPRM          ;DONE
```

```
 4423  020402  005722                          TST     (R2)+           ;BUMP THE POINTER
 4424  020404  000773                          BR      1$
 4425  020406  012022                  2$:     MOV     (R0)+,(R2)+     ;MOVE THIS VARIABLE INTO THE
 4426  020410  000771                          BR      1$              ;COMMON AREA
 4427  020412  013716  001372          OPNPRM: MOV     @#PRM,(SP)      ;GET THE VARIABLE INDICATOR
 4428  020416  005004                          CLR     R4              ;ZERO THE INDEX
 4429  020420  006216                  1$:     ASR     (SP)            ;CHECK FOR A VARIABLE
 4430  020422  103403                          BCS     3$              ;GO GET IT
 4431  020424  001772                          BEQ     OPNPRM          ;OUT OF VARIABLES
 4432  020426  005724                  2$:     TST     (R4)+           ;UPDATE THE INDEX
 4433  020430  000773                          BR      1$
 4434  020432  104401  002043          3$:     TYPE    ,BLNKS2         ;TYPE SPACES
 4435  020436  016437  001446  020446          MOV     PRMMSG(R4),4$   ;TYPE THE NAME OF THIS VARIABLE
 4436  020444  104401                          TYPE
 4437  020446  000000                  4$:     .WORD   0
 4438  020450  104401  021406                  TYPE    ,MSG.EQ         ;TYPE "="
 4439  020454  016446  001374                  MOV     RPT(R4),-(SP)   ;PUT RPT(R4) ON THE STACK
 4440  020460  004737  013220                  JSR     PC,$SB2O        ;CONVERT RPT(R4)
 4441  020464  004737  013450                  JSR     PC,$SUPRS       ;TYPE RPT(R4)
 4442  020470  104411                          RDLIN
 4443  020472  012601                          MOV     (SP)+,R1        ;READ AN ASCIZ STRING
 4444  020474  004037  021014                  JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
 4445  020500  020432                          3$                      ;ILLEGAL CHARACTER
 4446  020502  020432                          3$                      ;CARRIAGE RETURN
 4447  020504  020550                          7$                      ;"/"
 4448  020506  020514                          5$                      ;"-"
 4449  020510  020522                          6$                      ;"'"
 4450  020512  020432                          3$                      ;DIGIT 0-9
 4451  020514  105711                  5$:     TSTB    (R1)            ;"CR"?
 4452  020516  001345                          BNE     3$              ;NO--STAY ON THIS VARIABLE
 4453  020520  000742                          BR      2$              ;YES--MOVE TO NEXT VARIABLE
 4454  020522  105711                  6$:     TSTB    (R1)            ;IS THERE A "CR" AFTER THE PERIOD?
 4455  020524  001002                          BNE     64$             ;NO
 4456  020526  000137  020604                  JMP     OPN.N2          ;YES--GO CLOSE THIS TEST
 4457  020532  122721  000056          64$:    CMPB    #'.,(R1)+       ;DOUBLE PERIOD?
 4458  020536  001335                          BNE     3$              ;NO--GO ASK FOR THIS VARIABLE
 4459  020540  105711                          TSTB    (R1)            ;YES--IS A "CR" AFTER THE DOUBLE PERIOD?
 4460  020542  001333                          BNE     3$              ;NO--ASK FOR THIS VARIABLE AGAIN
 4461  020544  000137  020622                  JMP     OPN.X2          ;YES--CLOSE ALL TEST
 4462  020550                          7$:
 4463  020550  016402  001434                  MOV     PRMLMT(R4),R2   ;UPPER LIMIT OF INPUT
 4464  020554  004037  021070                  JSR     R0,@#CK.DIG     ;CHECK THE DIGIT(S)
 4465  020560  020432                          3$                      ;ILLEGAL INPUT
 4466  020562  020432                          3$                      ;INPUT TO LARGE
 4467  020564  020572                          8$                      ;TERMINATED WITH A ","
 4468  020566  020600                          OPN.N1                  ;TERMINATED WITH A ";"
 4469  020570  020616                          OPN.X1                  ;TERMINATED WITH A "."
 4470  020572  010264  001374          8$:     MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
 4471  020576  000713                          BR      2$              ;MOVE TO NEXT VARIABLE
 4472  020600  010264  001374          OPN.N1: MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
 4473  020604  005726                  OPN.N2: TST     (SP)+           ;CLEAN OFF THE STACK
 4474  020606  004737  020660                  JSR     PC,CLOSE        ;CLOSE THIS TEST
 4475  020612  000137  020270                  JMP     OPN.1           ;GO OPEN THE NEXT TEST
 4476  020616  010264  001374          OPN.X1: MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
 4477  020622  005726                  OPN.X2: TST     (SP)+           ;CLEAN OFF THE STACK
 4478  020624  004737  020660          1$:     JSR     PC,CLOSE        ;CLOSE THIS TEST
```

```
 4479  020630  005725                2$:     TST     (R5)+           ;UPDATE THE INDEX
 4480  020632  020527  000034                CMP     R5,#16*2        ;INDEX TO BIG?
 4481  020636  002404                        BLT     3$              ;NO--BRANCH
 4482  020640  104413                        RESREG                  ;YES, RESTORE R0-R5
 4483  020642  104413                        RESREG                  ;YES, RESTORE R0-R5
 4484  020644  000137  017706                JMP     GT,PR2          ;GO TO EXIT
 4485  020650  036503  001324        3$:     BIT     BITS(R5),R3     ;IS THIS TEST OPEN FOR CHANGE?
 4486  020654  001363                        BNE     1$              ;YES--GO CLOSE IT
 4487  020656  000764                        BR      2$              ;NO--MOVE TO NEXT TEST
 4488  020660  104412                CLOSE:  SAVREG                  ;SAVE R0-R5
 4489  020662  012700  001372                MOV     #PRM,R0         ;"FROM" ADDRESS
 4490  020666  016501  001406                MOV     PRMPT(R5),R1    ;"TO" ADDRESS
 4491  020672  012002                        MOV     (R0)+,R2        ;"FROM" INDICATOR
 4492  020674  012103                        MOV     (R1)+,R3        ;"TO" INDICATOR
 4493  020676  012704  000001                MOV     #1,R4           ;TEST BIT START A "RPT"
 4494  020702  030402                1$:     BIT     R4,R2           ;PARAMETER TO BE MOVED?
 4495  020704  001403                        BEQ     2$              ;NO--BRANCH
 4496  020706  030403                        BIT     R4,R3           ;A PLACE TO PUT IT?
 4497  020710  001404                        BEQ     3$              ;NO--BRANCH
 4498  020712  011011                        MOV     (R0),(R1)       ;YES--MOVE "FROM" TO "TO"
 4499  020714  030403                2$:     BIT     R4,R3           ;"TO" PARAMETER?
 4500  020716  001401                        BEQ     3$              ;NO--BRANCH
 4501  020720  005721                        TST     (R1)+           ;YES--UPDATE THE POINTER
 4502  020722  005720                3$:     TST     (R0)+           ;UPDATE FROM POINTER
 4503  020724  006304                        ASL     R4              ;POSITION THE TEST BIT
 4504  020726  032704  002000                BIT     #BIT10,R4       ;DONE?
 4505  020732  001763                        BEQ     1$              ;NO--BRANCH
 4506  020734  104413                        RESREG                  ;YES, RESTORE R0-R5
 4507  020736  000207                        RTS     PC              ;RETURN
 4508                                ;;*****************************************************************
 4509                                ;SBTTL  CK.OCT -- CHECK FOR OCTAL CHARACTER
 4510                                ;*THIS ROUTINE IS USED TO CHECK IF AN
 4511                                ;*ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
 4512                                ;CALL
 4513                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
 4514                                ;       JSR     R0,@#CK.OCT     ;CHECK THE CHARACTER
 4515                                ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
 4516                                ;       RETURN2                 ;CHARACTER IS IN R2 AS A
 4517                                ;                               ;OCTAL DIGIT
 4518
 4519  020740  121127  000060        CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
 4520  020744  103407                        BLO     1$              ;YES -- BRANCH
 4521  020746  121127  000067                CMPB    (R1),#'7        ;GREATER THAN SEVEN?
 4522  020752  101004                        BHI     1$              ;YES -- BRANCH
 4523  020754  111102                        MOVB    (R1),R2         ;GET THE CHARACTER
 4524  020756  042702  177770                BIC     #^C7,R2         ;STRIP AWAY THE ASCII
 4525  020762  005720                        TST     (R0)+           ;ADJUST FOR RETURN
 4526  020764  000200                1$:     RTS     R0              ;RETURN
 4527
 4528                                ;;*****************************************************************
 4529                                ;SBTTL  CK.DEC -- CHECK FOR DECIMAL CHARACTER
 4530                                ;*THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
 4531                                ;*AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
 4532                                ;CALL
 4533                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
 4534                                ;       JSR     R0,@#CK.DEC     ;CHECK THE CHARACTER
```

```
                                ;       RETURN1                 ;NOT BETWEEN 0 AND 9
                                ;       RETURN2                 ;BETWEEN 0 AND 9
                                ;                               ;R2 = DIGIT

020766  121127  000060          CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
020772  103407                          BLO     1$              ;YES -- BRANCH
020774  121127  000071                  CMPB    (R1),#'9                ;GREATER THAN NINE?
021000  101004                          BHI     1$              ;YES -- BRANCH
021002  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
021004  042702  000060                  BIC     #'0,R2          ;STRIP AWAY THE ASCII
021010  005720                          TST     (R0)+           ;ADJUST FOR RETURN
021012  000200                  1$:     RTS     R0              ;RETURN

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

021014  105711                  CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
021016  001420                          BEQ     4$              ;YES -- BRANCH
021020  121127  000057                  CMPB    (R1),#'/        ;"SLASH"?
021024  001414                          BEQ     3$              ;YES -- BRANCH
021026  121127  000054                  CMPB    (R1),#',        ;"COMMA"?
021032  001410                          BEQ     2$              ;YES -- BRANCH
021034  121127  000056                  CMPB    (R1),#'.        ;"PERIOD"?
021040  001404                          BEQ     1$              ;YES -- BRANCH
021042  004037  020766                  JSR     R0,@#CK.DEC     ;"DIGIT"?
021046  000406                          BR      5$              ;NO -- BRANCH
021050  005720                          TST     (R0)+           ;DIGIT BETWEEN 0-9
021052  005720                  1$:     TST     (R0)+           ;PERIOD
021054  005720                  2$:     TST     (R0)+           ;COMMA
021056  005720                  3$:     TST     (R0)+           ;SLASH
021060  005720                  4$:     TST     (R0)+           ;CARRIAGE RETURN
021062  005201                          INC     R1              ;MOVE POINTER TO NEXT CHARACTER
021064  011000                  5$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
021066  000200                          RTS     R0              ;RETURN

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

```
                                        ;       RETURN  ADR2            ;INPUT NUMBER TO LARGE -- R2=?
                                        ;       RETURN  ADR3            ;"COMMA" -- R2 = NUMBER
                                        ;       RETURN  ADR4            ;"PERIOD" -- R2 = NUMBER
                                        ;       RETURN  ADR5            ;"PERIOD-PERIOD" -- R2 = NUMBER
4596  021070  010446                    CK.DIG: MOV     R4,-(SP)        ;SAVE R4
4597  021072  010346                            MOV     R3,-(SP)        ;SAVE R3
4598  021074  010246                            MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
4599  021076  005002                            CLR     R2              ;START WITH 0
4600  021100  005003                            CLR     R3
4601  021102  005004                            CLR     R4
4602  021104  004037  021014                    JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
4603  021110  021234                            8$                      ;ILLEGAL CHARACTER
4604  021112  021234                            8$                      ;CARRIAGE RETURN
4605  021114  021234                            8$                      ;"/"
4606  021116  021234                            8$                      ;"-"
4607  021120  021234                            8$                      ;"'"
4608  021122  021124                            1$                      ;DIGIT 0-9
4609  021124  006303                    1$:     ASL     R3              ;*2
4610  021126  010346                            MOV     R3,-(SP)        ;SAVE *2
4611  021130  006303                            ASL     R3              ;*4
4612  021132  006303                            ASL     R3              ;*8
4613  021134  062603                            ADD     (SP)+,R3        ;(*8)+(*2)=*10.
4614  021136  060203                            ADD     R2,R3           ;UPDATE THE INPUT NUMBER
4615  021140  004037  021014                    JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
4616  021144  021234                            8$                      ;ILLEGAL CHARACTER
4617  021146  021234                            8$                      ;CARRIAGE RETURN
4618  021150  021234                            8$                      ;"/"
4619  021152  021162                            3$                      ;"-"
4620  021154  021160                            2$                      ;"'"
4621  021156  021124                            1$                      ;DIGIT 0-9
4622  021160  005724                    2$:     TST     (R4)+           ;"PERIOD"
4623  021162  005724                    3$:     TST     (R4)+           ;"COMMA"
4624  021164  004037  021014                    JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
4625  021170  021234                            8$                      ;ILLEGAL CHARACTER
4626  021172  021224                            6$                      ;CARRIAGE RETURN
4627  021174  021234                            8$                      ;"/"
4628  021176  021234                            8$                      ;"-"
4629  021200  021204                            4$                      ;"'"
4630  021202  021214                            5$                      ;DIGIT 0-9
4631  021204  005724                    4$:     TST     (R4)+           ;"PERIOD-PERIOD"
4632  021206  105711                            TSTB    (R1)            ;"CR"?
4633  021210  001405                            BEQ     6$              ;YES--BRANCH
4634  021212  000410                            BR      8$
4635  021214  126127  177776  000054    5$:     CMPB    -2(R1),#',      ;WAS CHARACTER BEFORE THE DIGIT A COMMA?
4636  021222  001004                            BNE     8$              ;NO--EXIT
4637  021224  020316                    6$:     CMP     R3,(SP)         ;INPUT TO LARGE?
4638  021226  101001                            BHI     7$              ;YES -- BRANCH
4639  021230  060400                            ADD     R4,R0           ;ADJUST RETURN ADDRESS
4640  021232  005720                    7$:     TST     (R0)+
4641  021234  010302                    8$:     MOV     R3,R2           ;NUMBER TO R2
4642  021236  005726                            TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
4643  021240  012603                            MOV     (SP)+,R3        ;RESTORE R3
4644  021242  012604                            MOV     (SP)+,R4        ;RESTORE R4
4645  021244  011000                            MOV     (R0),R0         ;GET RETURN ADDRESS
4646  021246  000200                            RTS     R0              ;RETURN
```

```
                                ;;*****************************************************************
                                .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
                                ;*THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
                                ;*AND FORMS AN OCTAL NUMBER IN R2
                                ;CALL
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                ;       JSR     R0,@#CK.NUM     ;GO FORM THE NUMBER
                                ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
                                ;       RETURN  ADR2            ;"COMMA"--R2=NUMBER
                                ;       RETURN  ADR3            ;"PERIOD"--R2=NUMBER
                                ;       RETURN  ADR4            ;"PERIOD-PERIOD"--R2=NUMBER

021250  010346                  CK.NUM: MOV     R3,-(SP)        ;SAVE R3
021252  005003                          CLR     R3              ;START NUMBER AT ZERO
021254  004037  020740                  JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
021260  000440                          BR      6$              ;NO--BRANCH
021262  005201                  1$:     INC     R1              ;MOVE TO NEXT CHARACTER
021264  006303                          ASL     R3              ;FOR THE OCTAL NUMBER IN R3
021266  103435                          BCS     6$              ;DON'T LET IT GET TO BIG
021270  006303                          ASL     R3
021272  103433                          BCS     6$
021274  006303                          ASL     R3
021276  103431                          BCS     6$
021300  060203                          ADD     R2,R3
021302  004037  020740                  JSR     R0,@#CK.OCT     ;IS THIS AN OCTAL DIGIT?
021306  000401                          BR      2$              ;NO--FIND OUT WHAT IT IS
021310  000764                          BR      1$              ;YES--MAKE IT PART OF THE NUMBER
021312  010302                  2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
021314  005003                          CLR     R3              ;START WITH ZERO INDEX
021316  004037  021014                  JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
021322  021362                          6$                      ;ILLEGAL CHARACTER
021324  021362                          6$                      ;CARRIAGE RETURN
021326  021362                          6$                      ;"/"
021330  021352                          5$                      ;"."
021332  021336                          3$                      ;"'"
021334  021362                          6$                      ;DIGIT 0-9
021336  005723                  3$:     TST     (R3)+           ;"PERIOD"
021340  121127  000056                  CMPB    (R1),#'.        ;"PERIOD-PERIOD"?
021344  001002                          BNE     5$              ;NO--BRANCH
021346  005201                          INC     R1              ;YES--ADVANCE THE POINTER
021350  005723                  4$:     TST     (R3)+           ;"PERIOD-PERIOD"
021352  005723                  5$:     TST     (R3)+           ;"COMMA"
021354  105711                          TSTB    (R1)            ;"CR"?
021356  001001                          BNE     6$              ;NO--BRANCH
021360  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
021362  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
021364  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
021366  000200                          RTS     R0              ;RETURN

                                .SBTTL  ASCIZ MESSAGES

021370  000122                  MSG.R:  .ASCIZ  /R/
021372  041506      000         MSG.FC: .ASCIZ  /FC/
021375      114  000103         MSG.LC: .ASCIZ  /LC/
```

```
4703  021400  041511  000             MSG.IC: .ASCIZ  /IC/
4704  021403  124     000113          MSG.TK: .ASCIZ  /TK/
4705  021406  000075                  MSG.EQ: .ASCIZ  /=/
4706  021410  005015  050122  030461  MRPCS:  .ASCIZ  <15><12>/RP11=/
4707  021416  000075
4708  021420  005015  050122  042526  MRPVEC: .ASCIZ  <15><12>/RPVEC=/
4709  021426  036503  000
4710  021431  015     051012  050120  MRPPRI: .ASCIZ  <15><12>/RPPRIO=/
4711  021436  044522  036517  000
4712
4713  021443  040     020040  020040  GRPH1:  .ASCII  /       0   10  20  30  40  50  60  70  80  90  100/<15><12>
4714  021450  020040  020060  020040
4715  021456  030061  020040  030062
4716  021464  020040  030063  020040
4717  021472  030064  020040  030065
4718  021500  020040  030066  020040
4719  021506  030067  020040  030070
4720  021514  020040  030071  020040
4721  021522  030061  006460  012
4722  021527  040     020040  020040          .ASCIZ  /       I---I---I---I---I---I---I---I---I---I---I/<15><12>
4723  021534  020040  026511  026455
4724  021542  026511  026455  026511
4725  021550  026455  026511  026455
4726  021556  026511  026455  026511
4727  021564  026455  026511  026455
4728  021572  026511  026455  026511
4729  021600  026455  026511  026455
4730  021606  006511  000012
4731                                          .EVEN
4732
4733
4734                                  .SBTTL  ERROR HEADER (EM) MESSAGES
4735
4736  021612  050122  030461  043040  EM1:    .ASCIZ  @RP11 FAILED TO RESPOND TO ADDRESSING@
4737  021620  044501  042514  020104
4738  021626  047524  051040  051505
4739  021634  047520  042116  052040
4740  021642  020117  042101  051104
4741  021650  051505  044523  043516
4742  021656  000
4743  021657  104     044522  042526  EM2:    .ASCIZ  @DRIVE UNSAFE@
4744  021664  052440  051516  043101
4745  021672  000105
4746  021674  051104  053111  020105  EM3:    .ASCIZ  @DRIVE OFFLINE@
4747  021702  043117  046106  047111
4748  021710  000105
4749  021712  051104  053111  020105  EM4:    .ASCIZ  @DRIVE ERROR DURING TIMING TEST@
4750  021720  051105  047522  020122
4751  021726  052504  044522  043516
4752  021734  052040  046511  047111
4753  021742  020107  042524  052123
4754  021750  000
4755  021751  116     020117  047111  EM5:    .ASCIZ  @NO INTERRUPT FROM POSITIONING AFTER 1 SECOND@
4756  021756  042524  051122  050125
4757  021764  020124  051106  046517
4758  021772  050040  051517  052111
```

```
 4759  022000  047511  044516  043516
 4760  022006  040440  052106  051105
 4761  022014  030440  051440  041505
 4762  022022  047117  000104
 4763  022026  044504  045523  042440  EM6:    .ASCIZ  @DISK ERROR AFTER POSITIONING@
 4764  022034  051122  051117  040440
 4765  022042  052106  051105  050040
 4766  022050  051517  052111  047511
 4767  022056  044516  043516     000
 4768  022063     103  051117  042522  EM7:    .ASCIZ  @CORRECT ATTN BIT NOT SET AFTER POSITIONING@
 4769  022070  052103  040440  052124
 4770  022076  020116  044502  020124
 4771  022104  047516  020124  042523
 4772  022112  020124  043101  042524
 4773  022120  020122  047520  044523
 4774  022126  044524  047117  047111
 4775  022134  000107
 4776  022136  051104  053111  020105  EM10:   .ASCIZ  @DRIVE OFFLINE AFTER POSITIONING@
 4777  022144  043117  046106  047111
 4778  022152  020105  043101  042524
 4779  022160  020122  047520  044523
 4780  022166  044524  047117  047111
 4781  022174  000107
 4782  022176  051447  041525  023501  EM11:   .ASCIZ  @'SUCA' NOT CORRECT AFTER POSITIONING@
 4783  022204  047040  052117  041440
 4784  022212  051117  042522  052103
 4785  022220  040440  052106  051105
 4786  022226  050040  051517  052111
 4787  022234  047511  044516  043516
 4788  022242     000
 4789  022243     104  051511  020113  EM12:   .ASCIZ  @DISK ERROR WHILE VERIFYING POSITION@
 4790  022250  051105  047522  020122
 4791  022256  044127  046111  020105
 4792  022264  042526  044522  054506
 4793  022272  047111  020107  047520
 4794  022300  044523  044524  047117
 4795  022306     000
 4796  022307     104  051511  020113  EM13:   .ASCIZ  @DISK POSITIONED TO WRONG CYLINDER@
 4797  022314  047520  044523  044524
 4798  022322  047117  042105  052040
 4799  022330  020117  051127  047117
 4800  022336  020107  054503  044514
 4801  022344  042116  051105     000
 4802  022351     116  020117  047111  EM14:   .ASCIZ  @NO INTERRUPT FROM I/O AFTER 1 SECOND@
 4803  022356  042524  051122  050125
 4804  022364  020124  051106  046517
 4805  022372  044440  047457  040440
 4806  022400  052106  051105  030440
 4807  022406  051440  041505  047117
 4808  022414  000104
 4809  022416  042523  045505  044440  EM15:   .ASCIZ  @SEEK INCOMPLETE@
 4810  022424  041516  046517  046120
 4811  022432  052105  000105
 4812  022436  051104  053111  020105  EM16:   .ASCIZ  @DRIVE NOT READY AFTER POSITIONING@
 4813  022444  047516  020124  042522
 4814  022452  042101  020131  043101
```

```
 4815  022460  042524  020122  047520
 4816  022466  044523  044524  047117
 4817  022474  047111  000107
 4818
 4819  022500  051104  053111  020105  EM17:   .ASCIZ  @DRIVE NOT READY/OFFLINE DURING TIMING TEST@
 4820  022506  047516  020124  042522
 4821  022514  042101  027531  043117
 4822  022522  046106  047111  020105
 4823  022530  052504  044522  043516
 4824  022536  052040  046511  047111
 4825  022544  020107  042524  052123
 4826  022552     000
 4827
 4828                          .SBTTL  DATA HEADER (DH) MESSAGES
 4829
 4830  022553     105  051122  050040  DH1:    .ASCIZ  @ERR PC  RPCS@
 4831  022560  020103  051040  041520
 4832  022566  000123
 4833  022570  042524  052123  021440  DH2:    .ASCIZ  @TEST #  ERR PC  DRIVE   RPDS    RPER    RPCS@
 4834  022576  020040  051105  020122
 4835  022604  041520  020040  051104
 4836  022612  053111  020105  020040
 4837  022620  050122  051504  020040
 4838  022626  020040  050122  051105
 4839  022634  020040  020040  050122
 4840  022642  051503     000
 4841  022645     124  051505  020124  DH11:   .ASCIZ  @TEST #  ERR PC  DRIVE FROM CYL DEST CYL SUCA    RPCA@
 4842  022652  020043  042440  051122
 4843  022660  050040  020103  042040
 4844  022666  044522  042526  043040
 4845  022674  047522  020115  054503
 4846  022702  020114  042504  052123
 4847  022710  041440  046131  051440
 4848  022716  041525  020101  020040
 4849  022724  051040  041520  000101
 4850  022732  042524  052123  021440  DH12:   .ASCIZ  @TEST #  ERR PC  DRIVE   RPDS    RPER    RPCS    RPCA    RPDA@
 4851  022740  020040  051105  020122
 4852  022746  041520  020040  051104
 4853  022754  053111  020105  020040
 4854  022762  050122  051504  020040
 4855  022770  020040  050122  051105
 4856  022776  020040  020040  050122
 4857  023004  051503  020040  020040
 4858  023012  050122  040503  020040
 4859  023020  020040  050122  040504
 4860  023026     000
 4861  023027     124  051505  020124  DH13:   .ASCIZ  @TEST #  ERR PC  DRIVE FROM CYL DEST CYL HDR CYL@
 4862  023034  020043  042440  051122
 4863  023042  050040  020103  042040
 4864  023050  044522  042526  043040
 4865  023056  047522  020115  054503
 4866  023064  020114  042504  052123
 4867  023072  041440  046131  044040
 4868  023100  051104  041440  046131
 4869  023106     000
 4870  023107     122  041520  020123  DH13A:  .ASCIZ  @RPCS    RPCA    RPDA    SUCA@
```

```
4871  023114  020040  051040  041520
4872  023122  020101  020040  051040
4873  023130  042120  020101  020040
4874  023136  051443  041525  000101
4875  023144  042524  052123  021440  DHSP:   .ASCIZ  @TEST #  ERR PC  DRIVE   RPDS    RPER    RPCS    RPWC@
4876  023152  020040  051105  020122
4877  023160  041520  020040  051104
4878  023166  053111  020105  020040
4879  023174  050122  051504  020040
4880  023202  020040  050122  051105
4881  023210  020040  020040  050122
4882  023216  051503  020040  020040
4883  023224  050122  041527     000
4884  023231     122  041120  020101  DHSPA:  .ASCIZ  @RPBA    RPCA    RPDA    RPM1    SUCA    SILO@
4885  023236  020040  051040  041520
4886  023244  020101  020040  051040
4887  023252  042120  020101  020040
4888  023260  051040  046520  020061
4889  023266  020040  051440  041525
4890  023274  020101  020040  051440
4891  023302  046111  000117
4892
4893
4894                                  .EVEN
4895
4896                                  .SBTTL  DATA TABLE (DT)
4897
4898  023306  001116  001266          DT1:    .WORD   $ERRPC,RPADR
4899  023312  001204  001116  001246  DT2:    .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS
4900  023320  002004  002006  002010
4901  023326  001204  001116  001246  DT11:   .WORD   $TMPO,$ERRPC,CHKDRV,CYL.CR,CYL.DS,$SUCA,$RPCA
4902  023334  001254  001262  002024
4903  023342  002016
4904  023344  001204  001116  001246  DT12:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPCA,$RPDA
4905  023352  002004  002006  002010
4906  023360  002016  002020
4907  023364  001204  001116  001246  DT13:   .WORD   $TMPO,$ERRPC,CHKDRV,CYL.CR,CYL.DS,$BDDAT
4908  023372  001254  001262  001126
4909  023400  002010  002016  002020  DT13A:  .WORD   $RPCS,$RPCA,$RPDA,$SUCA
4910  023406  002024
4911  023410  001204  001116  001246  DTSP:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPWC
4912  023416  002004  002006  002010
4913  023424  002012
4914  023426  002014  002016  002020  DTSPA:  .WORD   $RPBA,$RPCA,$RPDA,$RPM1,$SUCA,$SILO
4915  023434  002022  002024  002026
4916
4917                                  .SBTTL  DATA FORMAT (DF) TABLE
4918
4919  023442  000001                  DF1:    .WORD   1               ;NUMBER OF DATA HEADERS
4920  023444     002                          .BYTE   2               ;NUMBER OF WORDS IN DATA TABLE
4921  023445     000                          .BYTE   0               ;BOTH NUMBERS ARE OCTAL
4922
4923  023446  000001                  DF2:    .WORD   1
4924  023450     006                          .BYTE   6
4925  023451     000                          .BYTE   0
4926
```

```
4927  023452  000001          DF11:   .WORD   1
4928  023454     007                  .BYTE   7
4929  023455     000                  .BYTE   0
4930
4931  023456  000001          DF12:   .WORD   1
4932  023460     010                  .BYTE   8.
4933  023461     000                  .BYTE   0
4934
4935  023462  000002          DF13:   .WORD   2               ;2 DH'S TO BE TYPED
4936  023464     006                  .BYTE   6
4937  023465     000                  .BYTE   0
4938  023466  023107                  .WORD   DH13A
4939  023470     004                  .BYTE   4
4940  023471     000                  .BYTE   0
4941
4942  023472  000002          DFSP:   .WORD   2
4943  023474     007                  .BYTE   7
4944  023475     000                  .BYTE   0
4945  023476  023231                  .WORD   DHSPA
4946  023500     006                  .BYTE   6
4947  023501     000                  .BYTE   0
4948
4949                          .EVEN
4950
4951  023502  023144          ETSP:   .WORD   DHSP            ;MESSAGE BLOCK FOR SWITCH & TYPEOUTS
4952  023504  023410                  .WORD   DTSP
4953  023506  023472                  .WORD   DFSP
4954
4955  023510  000030          BUFR4:  .BLKW   24.             ;WORKING LOCATIONS FOR SEEK TIMER TEST
4956  023570  000030          BUFR5:  .BLKW   24.
4957  023630  000030          BUFR6:  .BLKW   24.
4958  023750  000030          BUFR7:  .BLKW   24.
4959  024010  000626          FRWSTR: .BLKW   406.
4960  025464  000626          RVSTR:  .BLKW   406.
4961          027140          BUFFER=.
4962
4963          000001          .END
```

```
AIE   = 020000          1238#   3824    4060
ATABIT  001314          1343#   1829    1861    3589    3666    3801    4292    4297
BITS    001324          1353#   1967    1996    2031    2068    2104    2180    2221    2271    2312    2362    2427    4331
                        4349    4352    4354    4404    4485
BIT0  = 000001          1169#
BIT00 = 000001          1159#   1169    1353
BIT01 = 000002          1158#   1168    1354
BIT02 = 000004          1157#   1167    1355
BIT03 = 000010          1156#   1166    1356
BIT04 = 000020          1155#   1165    1357
BIT05 = 000040          1154#   1164    1358
BIT06 = 000100          1153#   1163    1359    3643    4095
BIT07 = 000200          1152#   1162    1360
BIT08 = 000400          1151#   1161    1361
BIT09 = 001000          1150#   1160    1362    2658    3248
BIT1  = 000002          1168#
BIT10 = 002000          1149#   1363    2642    4504
BIT11 = 004000          1148#   1364    3255
BIT12 = 010000          1147#   1365
BIT13 = 020000          1146#   1366    2649
BIT14 = 040000          1145#   1367    3230
BIT15 = 100000          1144#   1368
BIT2  = 000004          1167#
BIT3  = 000010          1166#
BIT4  = 000020          1165#
BIT5  = 000040          1164#
BIT6  = 000100          1163#   3741    3847
BIT7  = 000200          1162#   4048
BIT8  = 000400          1161#
BIT9  = 001000          1160#
BL'KS1  002044          1551#   2550
BL  KS2 002043          1550#   2721    2728    2735    3991    4006    4434
BLNKS3  002042          1549#
BLNKS4  002041          1548#
BLNKS5  002040          1547#
BLNKS6  002037          1546#   2528
BLNKS7  002036          1545#   4004
BLNKS8  002035          1544#   4003
BLNKS9  002034          1543#
BLNK10  002033          1542#
BLNK11  002030          1541#
BPTVEC= 000014          1176#
BUFFER= 027140          3706*   3729    3754*   3755*   3756*   3757*   3758*   3759*   3760    3762    4961#
BUFR4   023610          3910    3992    4955#
BUFR5   023570          3911    3987    4956#
BUFR6   023650          3956    4007    4957#
BUFR7   023730          3957    3999    4958#
BUSADR  001224          1299#   1639*   1691*   1694*   1696*   4154    4156*
BYPASS  001244          1308#   1771*   1976*   2005*   2040*   2077*   2113*   2189*   2230*   2280*   2321*   2371*   2436*
                        3644    3742    3848    4096    4108
CHKDRV  001246          1309#   1902*   1906*   1912    2583    3800    4899    4901    4904    4907    4911
CKSWR = 104407          2638    2657    3229    3405#
CK.CHR  021014          4225    4276    4444    4563#   4602    4615    4624    4677
CK.DEC  020766          4539#   4571
CK.DIG  021070          4286    4464    4596#
CK.NUM  021250          4233    4660#
```

```
CK.OCT  020740          4317    4321    4335    4339    4519#   4662    4672
CLEAR = 000001          1187#   3572    3573    3611    3612    3641    3645    3661    3662    3739    3740    3750    3751
                        3819    3820    3845    3846    4053    4064    4112    4113
CLOSE   020660          4474    4478    4486#
CMPGAP  007322          2471    2517#
CNTRLC  001222          1298#   1692*   1697*   1765*   1852    1879
CR      = 000015        1264#   2802    2812
CRLF    = 000200        1085#   2773    2812
CSME    = 000040        1217#
CYL.CR  001254          1312#   1924*   3684*   4901    4907
CYL.DS  001262          1315#   3614*   4901    4907
DDISP = 177570          1091#   1270    1724
DECSK   004452          2052#   2055
DFLT    001460          1415#   1756
DFSP    023472          4942#   4953
DF1     023442          1584    4919#
DF11    023452          1640    4927#
DF12    023456          1647    4931#
DF13    023462          1654    4935#
DF2     023446          1591    1598    1605    1612    1619    1626    1633    1661    1668    1675    1682    4923#
DHSP    023144          4875#   4951
DHSPA   023231          4984#   4945
DH1     023353          1582    4830#
DH11    022645          1638    4841#
DH12    022732          1645    4850#
DH13    023027          1652    4861#
DH13A   023107          4870#   4938
DH2     022570          1589    1596    1603    1610    1617    1624    1631    1659    1666    1673    1680    4833#
DISPLA  001142          1270#   1724*   1732*   1975*   2004*   2039*   2076*   2112*   2188*   2229*   2279*   2320*   2370*
                        2435*   2641*   3267*
DISPRE  000174          1049#   1732
DPB     001364          1373#   1913*   1978*   1979*   1980*   1981*   2007*   2008*   2009*   2011*   2013*   2042*   2043*
                        2044*   2046*   2048*   2049    2051*   2053*   2054    2079*   2080*   2081*   2083*   2086*   2115*
                        2116*   2117*   2125*   2131*   2145*   2151*   2191*   2192*   2193*   2197*   2199*   2232*   2233*
                        2234*   2241*   2243*   2245*   2247*   2249*   2251*   2282*   2283*   2284*   2288*   2290*   2323*
                        2324*   2329*   2331*   2332    2334*   2335*   2339*   2373*   2374*   2380*   2382*   2385*   2438*
                        2439*   2445*   2448*   3613    3614    3615    3616    3618    3665    3669    3674    3722*   3723
                        3724    3725    3730    3760    3822    4058    4059    4114
DRVBAD  001252          1311#   1768*   1909    3801*
DRVMSK  001250          1310#   1903*   1904    1907*
DRVOK   003740          1905    1912#
DRVSEL  001232          1302#   1856*   1861*   1872    1904    1909*   2577    4197*   4269*   4292*   4297*
DRVSTA  001304          1332#   1816    1859    3567*   3568*   3569*   3570*   3578*   3586*
DRVTYP  001240          1301#   1779    1796    1829    3566*   3589*
DSELCT  015242          3629    3633    3637    3800#   3835    3839    3843    3855    4080    4091
DSKERR= 000001          1212#
DS JR   = 177570        1090#   1269    1723
DTSP    023410          4911#   4952
DTSPA   023426          4914#
DT1     023306          1583    4898#
DT11    023326          1639    4901#
DT12    023344          1646    4904#
DT13    023364          1653    4907#
DT13A   023400          4909#
DT2     023312          1590    1597    1604    1611    1618    1625    1632    1660    1667    1674    1681    4899#
EMTVEC= 000030          1179#   1710*   1711*
```

```
EM1     021612          1581    4736#
EM10    022136          1630    4776#
EM11    022176          1637    4782#
EM12    022243          1644    4789#
EM13    022307          1651    4796#
EM14    022351          1658    4802#
EM15    022416          1665    4809#
EM16    022436          1672    4812#
EM17    022500          1679    4819#
EM2     021657          1588    4743#
EM3     021674          1595    4746#
EM4     021712          1602    4749#
EM5     021751          1609    4755#
EM6     022026          1616    4763#
EM7     022063          1623    4768#
EOP   = 000002          1213#
ERR   = 100000          1240#   3658    3747    4109
ERRVEC= 000004          1172#   1721    1722*   1733*   3235    3236*   3238*   3241*   4187    4188*   4190*   4192*
ETSP    023502          2695    4951#
EXIT0   004144          1976    1984#
EXIT1   004302          2005    2015#
EXIT10  006274          2321    2348#
EXIT11  006606          2371    2406#
EXIT12  007462          2436    2512    2540    2561#
EXIT2   004474          2040    2056#
EXIT3   004644          2077    2091#
EXIT4   005026          2113    2164#
EXIT5   005374          2189    2204    2207#
EXIT6   005626          2230    2240    2255#
EXIT7   006020          2280    2299#
FC      001376          1382#
FC1     001644          1445#   2011
FC11    001774          1505#   2380    2385    2394
FC2     001660          1453#   2046    2054
FC3     001674          1461#   2083
FC4     001710          1469#   2123    2131    2162
FC5     001724          1477#   2195    2205
FC6     001740          1485#   2236
FC7     001754          1493#   2286    2287    2295
FMTE  = 001000          1221#
FRWSTR  024010          2327    2379    2442    2478#   2488    2519    3908    3909    4959#
FUV   = 040000          1226#
GETADR  017226          1776    4150#
GETNUM  017430          4160    4167    4178    4218#
GNS   = ****** U        1048    1743    1806    1821    1827    1833    1839    1845    1868    1876    1897    1916    2343
                        2392    2399    2459    2463    2467    2484    2532    2556    2575    2581    2589    2606    3397
                        3398    3399    3400    3401    3403    3405    3406    3407    3408    3409    3967    3971    3975
                        3979    3997    4011    4272    4304    4408
GOTYPE  016100          3953    3964#
GRPHI   021443          2469    4713#
GTSWR = 104406          1751    3403#
GTTST1  017712          4296    4301#   4329    4336    4347    4351    4363    4369    4375    4381    4388    4401
GTTST2  017764          4316#   4372
GTTST3  020132          4357#
GTTST4  020134          4318    4333    4356    4358#
GTTST5  020206          4361    4376#
```

```
GTTST6  020236            4378    4385#
GT.PRM  017560            1854    4268#
GT.PR1  017562            4269#   4277    4278    4279    4280    4281    4287    4288
GT.PR2  017706            4298#   4313    4384    4391    4484
HOR     = 004000          1236#   3727
HE      = 040000          1239#
HNF     = 010000          1205#
HOMSEK= 000015            1193#   1978    3674    3823
HT      = 000011          1082#   2771    2812
IC      001402            1384#
IC1     001650            1447#
IC2     001664            1455#   2048    2053
IC3     001700            1463#   2085
IC4     001714            1471#   2141    2161
IC5     001730            1479#   2201    2202
IC6     001744            1487#   2253
IC7     001760            1495#   2292    2296
IDE     = 000100          1233#   3727
INHOR   001236            1300#
INCSK   004422            2047#   2050
IOTVEC= 000020            1177#   1708*   1709*
LC      001400            1383#
LC1     001646            1446#   1788*   2013
LC11    001776            1506#   1794*   2382    2401
LC2     001662            1454#   1789*   2049    2051
LC3     001676            1462#   1795*   1798*   2089
LC4     001712            1470#   1790*   2142    2144    2151
LC5     001726            1478#   1791*   2196    2203
LC6     001742            1486#   1792*   2237
LC7     001756            1494#   1793*   2293    2297
LF      = 000012          1083#   2806    2812
LPE     = 000200          1219#
MAXCYL  001236            1304#   1778*   1781*   1782    1783    1788    1789    1790    1791    1792    1793    1794    2332
                          2334    2345    2451    2498    2504    2506    2538
MODE    = 010000          1237#   3727
MODERR= 000400            1220#
MSCS    021410            4158    4706#
MSG.PRI 021431            4172    4710#
MSG.PVEC 021420           4165    4708#
MSG.EQ  021406            4438    4705#
MSG.FC  021372            1409    4701#
MSG.IC  021400            1411    4703#
MSG.LC  021375            1410    4702#
MSG.R   021370            1408    4700#
MSG.TK  021403            1412    4704#
MYSTAL  001302            1325#   2138    2158
NXC     = 020000          1225#
NXME    = 000004          1214#
NXS     = 004000          1223#
NXT     = 010000          1224#
OPNPRM  020412            4422    4427#   4431
OPNTST  020260            4374    4383    4390    4393#
OPN.CT  020264            4395#   4397*   4398    4402    4410
OPN.M1  020600            4468    4472#
OPN.M2  020604            4456    4473#
OPN.X1  020616            4469    4476#
```

```
OPN.X2  020622          4461    4477#
OPN.1   020270          4397#   4405    4475
OPN.2   020312          4396    4399    4402#
OUTADR  001230          1301#
PC     =X000007         1103#   1745#   1764#   1776#   1777#   1854#   1983#   2012#   2014#   2047#   2052#   2084#   2087#
                        2126#   2132#   2146#   2152#   2198#   2200#   2242#   2244#   2246#   2248#   2250#   2252#   2289#
                        2291#   2326#   2329#   2337#   2376#   2381#   2383#   2386#   2395#   2396#   2402#   2403#   2441#
                        2446#   2449#   2489#   2535#   2559#   2598#   2601#   2612#   2617    2651#   2738#   2780#   2787#
                        2794#   2808#   2810#   2985#   2999#   3060#   3105#   3364#   3424#   3427#   3465#   3470#   3515#
                        3550#   3593#   3625#   3648#   3654#   3672#   3676#   3687#   3737#   3745#   3767#   3790#   3831#
                        3851#   3858#   3878#   4076#   4103#   4107#   4122#   4129#   4140#   4191#   4299#   4394#   4440#
                        4441#   4474#   4478#   4507#
PIRQ   = 177772         1089#
PIRQVE= 000240          1183#
PLOT    006760          2456#
PLTPT   007422          2489    2535    2545#
PRM     001372          1380#   4418    4427    4489
PRMLMT  001434          1401#   1782#   1783#   4463
PRMMSG  001446          1408#   4435
PRMPT   001406          1388#   4416    4490
PRM0    001632          1388    1438#   1757
PRM1    001640          1389    1443#
PRM10   001764          1396    1499#
PRM11   001770          1397    1503#
PRM12   002000          1398    1509#
PRM2    001654          1390    1451#
PRM3    001670          1391    1459#
PRM4    001704          1392    1467#
PRM5    001720          1393    1475#
PRM6    001734          1394    1483#
PRM7    001750          1395    1491#
PROG   = 002000         1222#
PR0    = 000000         1106#
PR1    = 000040         1107#
PR2    = 000100         1108#
PR3    = 000140         1109#
PR4    = 000200         1110#
PR5    = 000240         1111#
PR6    = 000300         1112#
PR7    = 000340         1113#   1766
PS     = 177776         1086#   1087    1767#   3620#
PSW    = 177776         1087#   3607#   3645#   3732#   3743#   3803#   3826#   3850#   4062#   4099#
PWRVEC= 000024          1178#
RDCHR  = 104410         3160    3406#
RDLIN  = 104411         3407#   4223    4274    4306    4442
RDSEK  = 000005         1189#
RDY    = 000200         1234#   3639
READ   = 000017         1194#   3726
RESREG= 104413          2737    3409#   3469    3592    3686    3766    3857    4023    4298    4400    4482    4483    4506
RESTAR  003716          1906#   2618
RESTOR  015264          2326    2376    2441    3654    3676    3815#   4107
RESTR1  015500          3821    3849#
RFSVEC= 000010          1173#
RPDUR   001266          1319#   1923    1977    2006    2041    2078    2114    2190    2231    2281    2322    2372    2437
                        3571    3608    3647    3707    3746    3778    3818    4042    4084    4157    4189    4898
RPBA   = 000010         1531#   3729#   3783
```

```
RPCA  = 000012          1532#  1924   3615*  3784   4058*
RPCS  = 000004          1529#  3572*  3573*  3575*  3611*  3612*  3613*  3618*  3619*  3641*  3642*  3661*  3662*
                        3731*  3739*  3740*  3750*  3751*  3781   3819*  3820*  3825*  3845*  3846*  4061*  4093*
                        4094*  4109   4112*  4113*  4114*
RPDA  = 000C14          1533#  3616*  3708   3730*  3785   4047   4050   4051   4063   4064   4068   4069   4098
RPDS  = 000000          1527#  3576   3580   3582   3584   3587   3779   4057*  4083
RPER  = 000002          1528#  3780
RPINIT  013612          1777   3562*
RPM1  = 000016          1534#  3786
RPM2  = 000020          1535#
RPM3  = 000022          1536#
RPPRIO  001272          1321#  3565   3607
RPT     001374          1381#  4439   4470*  4472*  4476*
RPT0    001634          1439#  1970
RPT1    001642          1444#  1999
RPT10   001766          1500#  2315
RPT11   001772          1504#  2365
RPT12   002002          1510#  2430
RPT2    001656          1452#  2034
RPT3    001672          1460#  2071
RPT4    001706          1468#  2107
RPT5    001722          1476#  2183
RPT6    001736          1484#  2224
RPT7    001752          1492#  2274
RPVEC   001270          1320#  3563   3617*  3721*  3821*  4056*
RPVC  = 000006          1530#  3728*  3782
RSTRT1  003674          1902#  1911
RSTRT2  003706          1904#  1908
RSTRT3  003714          1905#
RVSTR   025464          2336   2378   3954   3955   4960#
R0    =%000000          1094#  1735*  1736*  1737   1756*  1759   1760   1857*  1859   1861   1862*  1871*  1889
                        1899*  1923*  1924   1977*  2006*  2041*  2078*  2114*  2129*  2135*  2149*  2155*  2190*
                        2231*  2281*  2322*  2346*  2372*  2404*  2437*  2477*  2480   2486   2490   2492*  2494
                        2496*  2500   2502*  2504   2508*  2510*  2517*  2525   2537*  2538   2609*  2612   2680*
                        2681*  2682*  2683*  2684*  2685*  2686*  2687   2695*  2696   2701   2704*  2705   2709
                        2710   2729   2763   2764*  2765   2768*  2904   2914*  2918   2934   2935   2948*  3290
                        3315*  3333   3346*  3348*  3349*  3355*  3357*  3359   3363*  3374   3375*  3376   3377*
                        3378*  3379*  3380*  3445*  3446   3464*  3468*  3503   3504*  3505   3507   3509*  3510
                        3513*  3529   3532*  3535*  3539*  3542*  3545   3549*  3571*  3572*  3573*  3575*  3576
                        3580   3582   3584   3587   3608*  3611*  3612*  3613*  3615*  3616*  3618*  3619*  3623*
                        3641*  3642*  3647*  3661*  3662*  3682*  3684   3707*  3708   3728*  3729*  3730*  3731*
                        3735*  3739*  3740*  3746*  3750*  3751*  3777   3778*  3779   3780   3781   3782   3783
                        3784   3785   3786   3787   3788   3789*  3818*  3819*  3820*  3825*  3829*  3845*  3846*
                        3873   3875   3885   3890*  3905   3912*  3916   3918   3919*  3922   3928*  3938   3940
                        3943   3984*  3987   3992   3999   4007   4020   4021   4024*  4038   4042*  4045*  4047
                        4050   4051   4057*  4058*  4061*  4063   4064   4068   4069   4093*  4094*  4098   4112*
                        4113*  4114*  4139*  4157*  4159   4160*  4164*  4166   4167*  4171*  4173   4178*  4182*
                        4184*  4186*  4225*  4233*  4238   4239   4251   4252   4258*  4276*  4286*  4317*  4321*
                        4335*  4339*  4416*  4417   4425   4444*  4464*  4489*  4491   4498   4502   4525   4526*
                        4545   4546*  4571*  4573   4574   4575   4576   4577   4579*  4580*  4602*  4615*  4624*
                        4639*  4640   4645*  4646*  4662*  4672*  4677*  4692*  4694*  4695*
R1    =%000001          1095#  1757*  1758   1759*  1818*  1811   1816   1829   1848*  1858*  1863*  1872*  1885*
                        1886*  1893   2085*  2086   2088*  2089   2123*  2125   2141*  2142   2144*  2145   2161*
                        2162   2195*  2197   2201*  2203   2236*  2239   2241   2253*  2286*  2288   2296*  2297
                        2377*  2388*  2443*  2448   2486*  2487*  2488   2518*  2522*  2524*  2701*  2713   2717
                        2905   2918*  2919   2923   2947*  3291   3314*  3334   3337*  3340*  3350*  3356*  3360
```

```
                    3362* 3447* 3453* 3459* 3530  3533* 3536* 3540* 3541* 3543* 3544* 3546  3548*
                    3563* 3564* 3565* 3574* 3575  3578* 3586* 3589  3590* 3664* 3665* 3666  3800*
                    3801  3913* 3916  3919  3920* 3923  3929* 3938* 3940  3942  3943* 4187* 4190
                    4192  4224* 4242  4245  4247  4249  4275* 4283* 4293  4307* 4308  4310  4312
                    4320* 4323* 4332  4334* 4338* 4341* 4357* 4360  4364  4368  4370* 4371  4376*
                    4377  4379  4385* 4386  4443* 4451  4454  4457  4459  4490* 4492  4498* 4501
                    4519  4521  4523  4539  4541  4543  4563  4565  4567  4569  4578* 4632  4635
                    4664* 4685  4687* 4690
R2      =%000002    1096# 1758* 1760  1809* 1849* 1912* 1913  1918  2118* 2121* 2127  2133  2147
                    2153  2196* 2199  2202* 2205  2237* 2238* 2239  2287* 2290  2292* 2293  2295*
                    2444* 2445  2450* 2451  2519* 2534  2705* 2723* 2906  2917* 2921* 2924  2931*
                    2932* 2933  2938* 2946* 3292  3313* 3335  3341* 3344* 3348  3361* 3444* 3447
                    3448* 3454* 3455* 3460* 3461* 3531  3534* 3537* 3547* 3914* 3924* 3930* 3931
                    3944  3945* 3949* 4039  4047* 4048  4050* 4051  4053  4063* 4064  4066  4068*
                    4069  4071  4098* 4100  4115* 4116* 4117* 4118* 4119* 4125  4138* 4182  4186
                    4240  4253* 4254* 4255* 4256* 4257* 4285* 4292  4297  4319  4327* 4328  4330*
                    4331  4337  4345* 4346  4348* 4349  4352  4353  4354  4418* 4423  4425* 4463*
                    4470  4472  4476  4491* 4494  4523* 4524* 4543* 4544* 4598  4599* 4614  4641*
                    4671  4675*
R3      =%000003    1097# 2330  2338  2384  2387  2447  2488* 2534* 2547* 2709* 2719* 2846  2855*
                    2861* 2862* 2865* 2870* 2871* 2872  2881* 2907  2915* 2916* 2930* 2933* 2942*
                    2943* 2945* 3155  3157* 3158  3161* 3162  3169* 3170  3172  3180  3184  3186*
                    3192  3194  3196* 3199* 3233  3312* 3452* 3457* 3463* 3464  3915* 3921* 3926
                    3937* 3939* 3946* 3950* 4043* 4073* 4074  4102* 4121  4123  4124* 4125* 4128
                    4130  4131* 4158* 4165* 4172* 4218  4315* 4366* 4373  4382  4389  4404  4485
                    4492* 4496  4499  4597  4600* 4609* 4610  4611* 4612* 4613* 4614* 4637  4641
                    4643* 4660  4661* 4665* 4667* 4669* 4671* 4675  4676* 4684  4688  4689  4692
                    4693*
R4      =%000004    1098# 2327* 2330* 2336* 2338* 2378* 2387* 2442* 2447* 2710* 2711* 2847  2849*
                    2850* 2851* 2852  2853* 2867  2869* 2877* 2880* 3294  3311* 3450* 3453  3459
                    3461  3918* 3920  3931* 3932* 3933* 3934* 3935* 3936* 4159* 4164  4166* 4171
                    4173* 4174* 4175* 4176* 4177* 4184  4221  4240* 4253  4316* 4331* 4349  4352*
                    4354  4358  4362  4366  4428* 4432  4435  4439  4463  4470* 4472* 4476* 4493*
                    4494  4496  4499  4503* 4504  4596  4601* 4622  4623  4631  4639  4644*
R5      =%000005    1099# 2379* 2394* 2545  2546* 2551* 2558* 2703* 2707* 2726* 2733  2848  2854*
                    2856* 2858* 2859* 2860* 2861  2879* 2908  2910* 2912* 2919* 2923* 2938  2944*
                    3295  3310* 3451* 3455  3462  3907* 3952* 3985* 4001  4015* 4018  4319* 4324*
                    4325* 4326* 4327  4337* 4342* 4343* 4344* 4345  4402* 4403* 4404  4416  4479
                    4480  4485  4490
R6      =%000006    1100# 1102  1702* 1703* 1704
R7      =%000007    1101# 1103
SAVREG= 104412      2677  3408# 3443  3562  3606  3701  3815  3906  4268  4393  4488
SAVRP   015134      3625  3648  3737  3745  3777# 3831  3851  4076  4103
SEC.RD  001260      1314# 3723*
SEEK  = 000011      1191# 2007  2042  2079  2115  2191  2232  2282  2323  2373  2438
SEEKS   014006      1983  2012  2014  2047  2052  2084  2087  2126  2132  2146  2152  2198  2200
                    2242  2244  2246  2248  2250  2252  2289  2291  3606#
SEEKS0  014034      3610  3611#
SEEKS1  014264      3617  3645#
SEEKS2  014350      3650  3658#
SEEKS3  014432      3663  3670  3672#
SEEKS4  014470      3656  3673  3675  3678  3680#
SILO  = 000026      1538# 3788
SKDON   017142      4110  4115#
SMGRP   007152      2477#
SORT    015630      2346  2404  3905#
```

```
SP     =X000006          1102#  1706*  1721*  1729*  1733   1811*  1889*  1918*  1982*  2010*  2045*  2082*  2122*
                         2194*  2235*  2285*  2325*  2375*  2394*  2401*  2440*  2480*  2498*  2499*  2500   2525*
                         2545*  2558   2583*  2591*  2646   2660*  2663*  2713*  2717*  2763*  2764   2765*  2767
                         2768   2769*  2771   2773   2775   2781   2783*  2785*  2793*  2797   2801   2802   2806
                         2838*  2839   2840   2841*  2846*  2847*  2848*  2854   2879   2880   2881   2882*  2883*
                         2904*  2905*  2906*  2907*  2908*  2909*  2910   2913*  2926   2928*  2930   2940   2942
                         2944   2945   2946   2947   2948   2950*  2951*  2994*  2995*  2996   3000   3002   3011
                         3013   3016   3019*  3020*  3021   3026   3028   3030*  3031   3047*  3048*  3049   3059
                         3066*  3069*  3070*  3074*  3075*  3077   3080*  3087   3090*  3094   3096   3098   3099*
                         3106   3108   3110*  3111   3113*  3114*  3115*  3116*  3117*  3132*  3133*  3134*  3135*
                         3136*  3142*  3155*  3156*  3161   3164   3168*  3175   3179*  3198   3199   3200*  3201*
                         3202*  3235*  3238   3240   3241   3263   3264   3268*  3290*  3291*  3292*  3293*  3294*
                         3295*  3296*  3297*  3298*  3299*  3306*  3307*  3308*  3309*  3310   3311   3312   3313
                         3314   3315   3333*  3334*  3335*  3336*  3337   3339*  3341   3343*  3345*  3351*  3353
                         3358   3359*  3360*  3361   3362   3363   3374*  3375   3385*  3386*  3422   3423*  3425*
                         3426*  3444   3446*  3503*  3504   3513   3514*  3529*  3530*  3531*  3547   3548   3549
                         3609*  3621   3646*  3685   3702*  3708*  3709*  3710*  3711*  3712*  3713*  3714*  3715
                         3718*  3720*  3722   3725*  3726*  3727*  3731   3733   3744*  3764*  3765   3777*  3789
                         3816*  3822*  3823*  3824*  3825   3827   3849*  3856   3873*  3875*  3879*  3880*  3881*
                         3882*  3884*  3886*  3889   3987*  3992*  3999*  4007*  4014   4017   4038*  4039*  4040*
                         4059*  4060*  4061   4097*  4120*  4121*  4123*  4124   4127*  4128   4130*  4131   4137
                         4138   4139   4193   4221*  4224   4275   4307   4410*  4417*  4420*  4427*  4429*  4439*
                         4443   4473   4477   4596*  4597*  4598*  4610*  4613   4637   4642   4643   4644   4660*
                         4693
SPSAV    001242          1307#  3621*  3646   3733*  3744   3827*  3849
SRTINT   002724          1762   1766#
STACK =  001100          1077#  1706   1982   2010   2045   2082   2122   2194   2235   2285   2325   2375   2440
STALL    015546          2129   2135   2149   2155   3682   3875#
STALLA   015542          3623   3735   3829   3873#  4045
STALLB   015574          3874   3877   3881#
STALLG   001300          1324#  3622*  3624   3734*  3736   3828*  3830   4044*  4046
STALLO   001274          1322#  2124*  2130   2136   2137*  2138   2140*  2150   2156   2157*  2158   2160*
STALLI   001276          1323#  3683
START    002302          1693   1698#
START1   002246          1066   1691#
START2   002270          1068   1696#
START3   002236          1070   1689#
START4   002260          1072   1694#
START5   002710          1763#  3001   3084
START6   003432          1786   1802   1851#
STKLMT=  177774          1088#
STRT1A   002252          1690   1692#
STRT2A   002274          1695   1697#
STR20X   001264          1316#  1698*  1784   1851*
SUCA   = 000024          1537#  3684   3787
SUFU   = 001000          1202#  3576   3626   3832   4077
SUOL   = 040000          1207#  3580   3630   3836   4085
SURDY  = 100000          1208#  3584   3634   3840   4085
SURPU3= 020000           1206#  3587
SUSI   = 004000          1204#  3582   3649   3852   4104
SUSU   = 002000          1203#
SUMP   = 000400          1201#
SWP      001140          1269#  1704   1723*  1725   1731*  1749   1801   2119   2470   2642   2649   2654   2658
                         2692   3004   3043   3098*  3230   3248   3255   3655   3677   3680   3703   3876
SWREG    000176          1050#  1731   1749   3004   3043   3066
SWD    = 000001          1141#  3703
```

```
SW00  = 000001          1131#   1141
SW01  = 000002          1130#   1140
SW02  = 000004          1129#   1139
SW03  = 000010          1128#   1138    3680
SW04  = 000020          1127#   1137
SW05  = 000040          1126#   1136
SW06  = 000100          1125#   1135    2692
SW07  = 000200          1124#   1134    1801
SW08  = 000400          1123#   1133
SW09  = 001000          1122#   1132    3655    3677
SW1   = 000002          1140#   2119
SW10  = 002000          1121#
SW11  = 004000          1120#
SW12  = 010000          1119#
SW13  = 020000          1118#
SW14  = 040000          1117#
SW15  = 100000          1116#
SW2   = 000004          1139#   3876
SW3   = 000010          1138#
SW4   = 000020          1137#   2470
SW5   = 000040          1136#
SW6   = 000100          1135#
SW7   = 000200          1134#
SW8   = 000400          1133#
SW9   = 001000          1132#
TBITVE= 000014          1174#
TEST0   004134          1971    1982#
TEST1   004252          2000    2010#
TEST10  006120          2316    2325#
TEST11  006374          2366    2375#
TEST12  006706          2431    2440#
TEST2   004410          2035    2045#
TEST3   004602          2072    2082#
TEST4   004766          2108    2120    2122#
TEST5   005314          2184    2194#
TEST6   005502          2225    2235#
TEST7   005734          2275    2285#
TIMEE = 000020          1216#
TIMSEK  016476          2329    2337    2381    2383    2386    2446    2449    4038#
TIMSK1  016520          4041    4043#
TIMSK2  017036          4056    4097#
TK      001404          1385#
TKVEC = 000060          1181#   1766*   2981*   2982*
TK0     001636          1440#   1981
TK1     001652          1448#   2009
TK2     001666          1456#   2044
TK3     001702          1464#   2081
TK4     001716          1472#   2117
TK5     001732          1480#   2193
TK6     001746          1488#   2234
TK7     001762          1496#   2284
TPVEC = 000064          1182#
TRAPVE= 000034          1180#   1712*   1713*
TRK.RD  001256          1313#   3724*
TRTVEC= 000014          1175#
TSTNMS  001234          1303#   1755*   1967    1996    2031    2068    2104    2180    2221    2271    2312    2362    2427
```

```
                        4295*   4314*   4358*
TST0    004024          1966#   1972
TST1    004146          1969    1995#   2001
TST10   006022          2273    2311#   2317
TST11   006276          2314    2361#   2367
TST12   006610          2364    2426#   2432
TST2    004304          1998    2030#   2036
TST3    004476          2033    2067#   2073
TST4    004646          2070    2103#   2109
TST5    005210          2106    2179#   2185
TST6    005376          2182    2220#   2226
TST7    005630          2223    2270#   2276
TYPOS = 104405          2481    2526    2718    3401#   3989    3993    4005    4008
TYPE  = 104401          1741    1804    1819    1825    1831    1837    1843    1847    1866    1874    1895    1914    1922
                        2341    2390    2397    2457    2461    2465    2469    2479    2482    2520    2528    2530    2536
                        2550    2554    2573    2579    2587    2604    2608    2644    2652    2689    2690    2698    2699
                        2708    2721    2725    2728    2730    2732    2735    2776    2873    2949    2998    3010    3064
                        3065    3068    3079    3089    3100    3119    3167    3173    3178    3182    3187    3188    3190
                        3193    3197    3397#   3511    3965    3969    3973    3977    3986    3990    3995    4003    4004
                        4006    4009    4219    4270    4302    4406    4415    4434    4436    4438
TYPERR  010072          2651    2677#
TYPOC = 104402          2592    2714    3067    3398#   4222
TYPON = 104404          3400#
TYPOS = 104403          1813    1890    1919    2584    3399#   4412
TYPTIM  016334          3984#
VERFY0  014550          3705    3706#
VERFY1  015124          3704    3752    3761    3765#
VERIFY  014524          3672    3701#
WCE   = 000010          1215#
WCKSEK= 000007          1190#
WPE   = 000100          1218#
WPV   = 100000          1227#
WRITE = 000013          1192#
WRTSEK= 000003          1188#
$AUTOB  001134          1266#   1753*   3057    3214
$BDADR  001122          1261#   4083*   4084*
$BDDAT  001126          1263#   3762*   4082*   4086    4907
$BELL   001212          1293#   1737    2644    2666    3010    3207
$CHARC  010544          2778*   2788*   2795    2804*   2809#
$CKSUM  011520          3043#   3405
$CMTAG  001100          1249#   1701    1702    1710    1714    1715
$CM1  = 000011          1281#   1282#   1283#   1284#   1285#   1286#   1287#   1288#   1289#   1290#
$CM2  = 000022          1281#   1282#   1283#   1284#   1285#   1286#   1287#   1288#   1289#   1290#
$CM3  = 000011          1279#   1281
$CM4  = 000001          1290#   1291#
$CNTLC  012441          2998    3079    3207#
$CNTLG  012453          3064    3209#
$CNTLU  012446          3089    3182    3208#
$CRLF   001217          1295#   1847    1922    2479    2520    2536    2652    2666    2689    2698    2708    2725    2732
                        2777    2812    3100    3187    3207    3986    4415
$DBLK   011212          2915    2949    2957#
$DB2D   013254          3424    3443#
$DECVL  013434          3445    3491#
$DOAGN  007722          2600    2610    2616#
$DTBL   011202          2918    2953#
$ENDAD  007712          1058    2612#
```

```
$ENDCT  007646          1870*  1888*  1901   2602#
$ENULL  007726          2608   2619#
$EOP    007464          1878   2429   2572#  3804   4199
$EOPCT  007640          1901#  2599#  2603   4198*
$ERFLG  001103          1252#  2639#  2666   3220   3244   3246   3252*  3271
$ERMAX  001115          1258#  1716#  1773*  3246   3266*  3271
$ERROR  007732          1710   2637#
$ERRPC  001116          1259#  2646*  2647*  2648   2666   4898   4899   4901   4904   4907   4911
$ERRTB  002046          1568#  2686
$ERTTL  001112          1256#  2591   2593*  2645*  2666
$ESCAP  001210          1292#  1715*  1770*  2661   2663   2666   3265*  3651*  3653*
$FILLC  001156          1277#  2781   2812
$FILLS  001155          1276#  2812
$GDADR  001120          1260#
$GDDAT  001124          1262#  4085*  4088
$GET42  007702          2609#
$GTSWR  011610          3065#  3403
$HD   = 000000          1023
$HINUM  013606          3533   3541   3546*  3551#
$ICNT   001104          1253#  1772*  3257*  3258   3260*  3270
$INTAG  001135          1267#  3062*  3081   3101   3214
$ITEMB  001114          1257#  2648*  2666   2681
$LF     001220          1296#  2666   2812   3197   3207
$LONUM  013610          3532   3539   3545*  3552#  3879
$LPADR  001106          1254#  1717*  1774*  1971*  2000*  2035*  2072*  2108*  2184*  2225*  2275*  2316*  2366*
                        2431*  3250*  3263*  3268   3270
$LPERR  001110          1255#  1718*  1775*  1972*  2001*  2036*  2073*  2109*  2185*  2226*  2276*  2317*  2367*
                        2432*  2660   3250   3264*  3270   3609   3610*  3657   3679   3685*  3702   3705*  3765*
                        3816   3817*  3856*  4040   4041*  4137*
$MAIL = ****** U        1735   1749   2654   2765   3263
$MNEW   012471          3068   3212#
$MSWR   012460          3065   3210#
$MUL  = 000003          4123#  4125   4130#  4132
$MULT   013022          3332#  4122   4129
$MXCNT  012724          3261   3270#
$NULL   001154          1275#  2783   2812
$NWTST= 000001          1956#  1958   1986#  1988   2017#  2019   2058#  2060   2093#  2095   2166#  2168   2209#
                        2211   2257#  2259   2301#  2303   2350#  2352   2409#  2411
$OCNT   010772          2845*  2874*  2887#
$OMODE  010774          2840*  2844*  2849   2852*  2863*  2889#
$OVER   012710          3231   3251   3259   3267#
$PASS   001100          1250#  2596*  2597*  2619
$QUES   001216          1294#  2666   2812   3119   3190   3207
$RAND   013510          3528#  3878
$RDCHR  012062          3132#  3406
$RDDEC= ****** U        3406
$RDLIN  012152          31[illegible]  3407
$RDOCT= ****** U        34[illegible]
$RDSZ = 000033          3[illegible]#
$REGAD  001160          1[illegible]#  1735
$REG0   001162          1[illegible]#  3908*  3912   3928   3954*
$REG1   001164          1282#  3909*  3913   3955*
$REG10  001202          1289#
$REG2   001166          1283#  3910*  3929   3956*
$REG3   001170          1284#  3911*  3930   3957*
$REG4   001172          1285#
```

```
$REGS   001174          1286#
$REG6   001176          1287#
$REG7   001200          1288#
$RESRE  012764          3305#   3409
$RPBA   002014          1518#   3783*   4914
$RPCA   002016          1519#   3784*   4901    4904    4909    4914
$RPCS   002010          1516#   3639    3658    3747    3781*   4899    4904    4909    4911
$RPDA   002020          1520#   3785*   4904    4909    4914
$RPDS   002004          1514#   3626    3630    3634    3649    3666    3779*   3832    3836    3840    3852    4077    4082
                        4104    4899    4904    4911
$RPER   002006          1515#   3780*   4899    4904    4911
$RPMI   002022          1521#   3786*   4914
$RPWC   002012          1517#   3782*   4911
$RTNAD  007724          2618#
$R2A  = ****** U        3410
$SAVRE  012726          3289#   3408
$SB2D   013220          2395    2402    3422#   4440
$SCOPE  012502          1708    3228#
$SETUP= 000107          1689#   1707    1708    1710    1712    1714    1715    1717    1746    2594    2638    2657    2665
                        3002    3007    3008    3038    3214    3229
$SILO   002026          1523#   3788*   4914
$STUP = 177777          1689#
$SUCA   002024          1522#   3669    3787*   4901    4909    4914
$SUPRS  013450          2396    2403    3503#   4441
$SVLAD  012662          3239    3262#
$SVPC = 000200          1056#   1061
$SWR  = 167000          1013#   1023    1028    1029    1030    1031    1032    1033    1034    1291    1292    1293    1714
                        1715    1717    1718    1978    2007    2042    2079    2115    2191    2232    2282    2323    2373
                        2438    2569    2595    2611    2617    2619    2623#   2629    2630    2631    2632    2633    2642
                        2649    2654    2658    2666    3221    3222    3223    3224    3225    3230    3242    3244    3245
                        3246    3253    3254    3255    3264    3267    3270
$SWREG= 000000          3225
$TIMES  001206          1291#   1714*   1769*   1970*   1999*   2034*   2071*   2107*   2183*   2224*   2274*   2315*   2365*
                        2430*   2595*   3253*   3258    3261*   3270
$TKB    001146          1272#   2962    2983    2994    3019    3047    3074
$TKCNT  011222          2963#   2978*   3008    3025*   3139    3141*
$TKINT  011232          1745    1764    2978#   2999    3060
$TKQEN= 011231          2967#   3033    3144
$TKQIN  011224          2964#   2979*   2980    3031*   3032*   3033    3035*
$TKQOU  011226          2965#   2980*   3142    3143*   3144    3146*
$TKQSR  011230          2966#   2979    3035    3146
$TKS    001144          1271#   2962    2984*   3015*   3017    3023*   3045    3061*   3071    3083*   3103*   3643*   3741*
                        3847*   4095*
$TKSRV  011302          2931    2994#
$TMPO   001204          1290#   2678*   2679*   4086*   4087*   4088    4899    4901    4904    4907    4911
$TN   = 000013          1013#   1023    1956    1966    1978#   1986    1995    2007#   2017    2030    2042#   2058    2067
                        2079#   2093    2103    2115#   2166    2179    2191#   2209    2220    2232#   2257    2270    2282#
                        2301    2311    2323#   2350    2361    2373#   2409    2426    2438#   4328    4346    4398
$TNPWR  013364          3450    3451    3471#
$TPB    001152          1274#   2801*   2812
$TPFLG  001157          1278#   2759    2812
$TPS    001150          1273#   2799    2812
$TRAP   013134          1712    3374#
$TRAP2  013156          3385#   3396
$TRP  = 000014          3389#   3398#   3399#   3400#   3401#   3402#   3403    3404#   3405    3406#   3407#   3408#   3409#
                        3410#
```

```
$TRPAD  013170          3379    3396#
$TSTNM  001102          1251#   1973#   1975    2002#   2004    2037#   2039    2074#   2076    2110#   2112    2186#   2188
                        2227#   2229    2277#   2279    2318#   2320    2368#   2370    2433#   2435    2594#   2641    2666
                        2679    3220    3262#   3267    3271
$TTYIN  012406          3157    3158    3170    3188    3202    3206#
$TYPBN= ****** U        3402
$TYPOS  010776          2903#   3401
$TYPE   010330          2759#   3389    3397
$TYPEC  010500          2780    2787    2794    2799#   2800    3105
$TYPEX  010546          2805    2807    2810#
$TYPOC  010574          2843#   3398
$TYPON  010610          2842    2845#   3400
$TYPOS  010550          2838#   3399
$XTSTR  012514          3233#
$$GET4= 000000          2611#
$OFILL  010773          2839#   2843#   2853    2888#
$OCAT=  ****** U        2651    3230
.       = 027140        1044#   1048#   1056    1057#   1059#   1061#   1064#   1248#   1297    1705    1717    1718    1774
                        1775    1834#   1840#   1846#   1869#   1877#   2344#   2393#   2460#   2468#   2576#   2619    2620#
                        2666    2812    2957#   2962    2966#   2967    2968#   3206#   3207    3214    3270    3271    3491#
                        3972#   4409#   4955#   4956#   4957#   4958#   4959#   4960#   4961
```

```
CHECK   1685#  4081
CKCHR   1685#  4225  4276  4444  4602  4615  4624  4677
CKDIG   1685#  4284  4462
CKNUM   1685#  4232
COMMEN     1#  1184#  1928
ENDCOM     1#  1184#  1950
ENDPAS  1685#  2604
ERROR   1078#  3628  3632  3636  3638  3652  3660  3668  3671  3738  3749  3763  3834  3838  3842
        3844   3854  4079  4090  4092  4106  4111  4194
ESCAPE     1#  1184#
GET <I     1#  1184#
GETS\      1#  1013#  1184#  1746
MORETM  1242#  1298
MORE.S  1685#  1966  1995  2030  2067  2103  2179  2220  2270  2311  2361  2426
MSG     1956#  1958  1986#  1988  2017#  2019  2058#  2060  2093#  2095  2166#  2168  2209#  2211  2257#
        2259   2301#  2303  2350#  2352  2409#  2411
MULT       1#  1184#  4120  4127
NEWTST     1#  1184#  1956  1986  2017  2058  2093  2166  2209  2257  2301  2350  2409
POP        1#  1184#  2944  3310  3361  3547
PUSH       1#  1184#  2903  3290  3332  3528
REPORT     1#  1184#
SCOPE   1079#  1984  2015  2056  2091  2164  2207  2255  2299  2348  2406  2561
SETPRI     1#  1184#  3135
SETTRA  3389#  3398  3399  3400  3401  3403  3405  3406  3407  3408  3409
SETUP      1#  1184#  1700
SET.TN  1685#  1967  1996  2031  2068  2104  2180  2221  2271  2312  2362  2427
SKIP       1#  1184#
SLASH      1#  1184#
SPACE   1184#
STARS      1#  1054  1184#  1244  1297  1956  1965  1986  1994  2017  2029  2058  2066  2093  2102
        2166   2178  2209  2219  2257  2269  2301  2310  2350  2360  2409  2425  2566  2625  2667
        2744   2815  2893  2961  3038  3053  3124  3148  3217  3274  3320  3368  3413  3432  3495
        3519   3554  3595  3689  3769  3792  3806  3860  3893  4026  4133  4135  4142  4201  4260
        4508   4528  4548  4582  4648
STIME   1685#  3622  3734  3828  4044
SWRSU      1#  1184#  1719#
TRMTRP  3389#
TYPBIN     1#  1184#
TYPBED  1685#  4439
TYPDEC     1#  1184#  2716
TYPEND  1685#  2573
TYPNAM     1#  1184#
TYPNUM     1#  1184#
TYPOCS     1#  1184#  1810  1889  1918  2583  4410
TYPOCT     1#  1184#  2591  2713  3066  4221
TYPTXT     1#  1184#  1741  1803  1819  1824  1831  1836  1842  1865  1874  1895  1914  2341  2390
        2397   2456  2461  2465  2482  2529  2553  2573  2579  2587  2604  3964  3969  3973  3977
        3995   4009  4270  4301  4406
$$CMRE  1242#  1281  1282  1283  1284  1285  1286  1287  1288  1289
$$CMTM  1242#  1290
$ ESCA     1#  1184#
$ EWT      1#  1184#  1956  1986  2017  2058  2093  2166  2209  2257  2301  2350  2409
$$SET   3389#  3398  3399  3400  3401  3403  3405  3406  3407  3408  3409
$$SKIP     1#  1184#
.EQUAT     1#  1013#  1074
.HEADE     1#  1013#
```

```
.KT11       1#
.SETUP      1#   1013#   1689
.SWRHI      1#   1013#   1024
.SWRLO   1034#   1035    1036    1037    1038    1039
.SACTI      1#   1013#   1052
.SAPTB      1#
.SAPTH      1#
.SAPTY      1#
.SASTA      1#
.SCATC      1#   1013#   1042
.SCMTA      1#   1013#   1242
.SDB2D      1#   1013#   3430
.SDB20      1#
.SDIV       1#
.SEOP       1#   1013#   2564
.SERRO      1#   1013#   2623
.SERRT      1#
.SMULT      1#   1013#   3318
.SPOWE      1#
.SRAND      1#   1013#   3517
.SRDDE      1#   1013#
.SRDOC      1#   1013#
.SREAD      1#   1013#   2959
.SR2A2      1#
.SSAVE      1#   1013#   3272
.S[illegible]2D      1#   1013#   3411
.SS620      1#
.SSCOP      1#   1013#   3215
.SSIZE      1#
.SSUPR      1#   1013#   3493
.STRAP      1#   1013#   3366
.STYPB      1#
.STYPD      1#   1013#   2891
.STYPE      1#   1013#   2742
.STYPO      1#   1013#   2813
.SWOCA      1#
.1170       1#
```

DZRPZB.CMB CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
ADC     3460  3540  3543
ADD     2048  2141  2201  2243  2247  2251  2253  2292  2296  2496  2502  2508  2510  2686  2769
        2941  2951  2923  3080  3090  3099  3348  3425  3459  3461  3539  3541  3542  3544  3714
        3764  4084  4125  4327  4345  4613  4614  4639  4671  4692
ASL     2098  2487  2683  2684  2685  3113  3114  3115  3378  3535  4174  4175  4176  4324  4325
        4326  4330  4342  4343  4344  4348  4403  4503  4609  4611  4612  4665  4667  4669
ASLB    1863  1907  2928
ASR     1885  1896  3710  3711  3712  3713  3754  3755  3756  3757  3758  3759  4116  4117  4118
        4119  4255  4256  4257  4420  4429
BCC     1887  1908  2929  3347
BCS     2712  4421  4430  4666  4668  4670
BEQ     1780  1797  1800  1817  1853  1882  1894  2120  2128  2134  2148  2154  2505  2578  2610
        2640  2643  2659  2662  2688  2694  2697  2702  2706  2720  2772  2807  2868  3005  3058
        3097  3112  3140  3176  3245  3249  3354  3506  3508  3577  3581  3585  3588  3627  3650
        3656  3659  3670  3675  3678  3681  3748  3761  3833  3853  3877  3941  3947  3953  3988
        4000  4002  4049  4067  4078  4089  4105  4110  4155  4196  4246  4313  4355  4361  4363
        4378  4380  4387  4405  4422  4431  4495  4497  4500  4505  4564  4566  4568  4570  4633
BGE     2050  2163  2206  2294  2298  2452  3259  4351  4399
BGT     2139  2159  2204  2240  2600  2875  2937  3029  3109  4075  4329  4347
BHI     3247  3716  4522  4542  4638
BIC     1909  2499  2597  2865  2995  3020  3030  3048  3075  3110  3709  3880  4087  4115  4524
        4544
BIS     2870  2871  2931  2932  3117  3463  3727  3824  4060  4352  4358  4366
BISB    1861  3589  3619  3801  4292  4297
BIT     1801  1904  1967  1996  2031  2068  2104  2119  2180  2221  2251  2312  2362  2427  2470
        2642  2649  2658  2692  3230  3248  3255  3576  3580  3582  3584  3587  3626  3630  3634
        3639  3649  3655  3658  3677  3680  3703  3747  3832  3836  3840  3852  3876  4048  4053
        4066  4071  4077  4100  4104  4109  4354  4404  4485  4494  4496  4499  4504
BITB    1829  2795  3666
BLE     1860  2055  2090  2143  2333  2491  2495  2501  2724  3917
BLO     1761  3171  3717  3883  4520  4540
BLOS    3159
BLT     2548  2786  2876  2920  2936  3027  3107  3456  4481
BMI     1818  2927
BNE     1705  1726  1738  1740  1748  1750  1785  1802  1830  1850  1864  1873  1880  1884  1905
        1968  1997  2032  2069  2105  2181  2222  2272  2313  2363  2389  2428  2471  2507  2523
        2539  2650  2727  2734  2766  2774  2782  2796  2803  2866  2925  2997  3003  3009  3014
        3022  3034  3044  3050  3078  3082  3088  3095  3102  3145  3163  3165  3181  3185  3195
        3231  3256  3352  3467  3538  3583  3631  3635  3640  3667  3704  3837  3841  3887  3925
        3927  3951  4019  4022  4052  4054  4065  4070  4072  4101  4243  4248  4250  4294  4309
        4311  4333  4365  4369  4372  4374  4383  4390  4452  4455  4458  4460  4486  4636  4686
        4691
BPL     2340  2552  2655  2760  2800  2864  2911  2941  3018  3046  3072  3338  3342  3591
BR      1690  1693  1695  1728  1742  1752  1762  1805  1820  1823  1826  1832  1835  1838  1841
        1844  1855  1867  1875  1896  1900  1911  1915  2254  2342  2391  2398  2458  2462  2466
        2483  2493  2497  2503  2509  2511  2521  2527  2531  2555  2574  2580  2588  2605  2715
        2722  2736  2762  2779  2789  2798  2805  2842  2857  2878  2922  2939  3012  3091  3118
        3120  3174  3183  3189  3191  3233  3239  3242  3251  3254  3458  3579  3663  3673  3719
        3752  3874  3838  3948  3966  3970  3974  3978  3956  4010  4013  4016  4161  4162  4163
        4168  4169  4170  4179  4180  4181  4183  4185  4241  4244  4271  4296  4303  4318  4322
        4336  4340  4356  4367  4375  4396  4407  4419  4424  4426  4433  4453  4471  4487  4572
        4634  4663  4673  4674
CLR     1691  1692  1696  1703  1714  1715  1736  1767  1768  1769  1770  1771  1808  1851  1856
        1857  1870  1871  1902  1979  1980  2008  2043  2080  2116  2118  2124  2140  2160  2192
        2233  2283  2324  2374  2439  2443  2444  2477  2478  2517  2593  2594  2595  2678  2680
        2703  2855  2914  2917  2978  3015  3061  3069  3070  3134  3135  3156  3179  3253  3265
```

```
        2336  3346  3452  3564  3566  3567  3568  3569  3570  3620  3645  3653  3664  3718  3732
        3743  3803  3826  3850  3881  3915  3932  3933  3934  3935  3936  3984  3985  4043  4062
        4099  4156  4197  4198  4269  4295  4314  4315  4316  4394  4428  4599  4600  4601  4661
        4676
CLRB    2778  2804  2943  3186  3196  3252  3468
CMP     1704  1725  1737  1749  1760  2049  2054  2089  2138  2142  2158  2162  2203  2205  2239
        2293  2297  2332  2451  2490  2494  2500  2504  2506  2538  2935  2996  3002  3004  3008
        3013  3021  3026  3028  3033  3043  3049  3077  3087  3094  3106  3108  3144  3158  3170
        3240  3258  3353  3462  3669  3715  3760  3889  3916  3940  4021  4051  4064  4069  4074
        4088  4193  4328  4346  4349  4398  4480  4637
CMPB    1683  2771  2773  2781  2802  2806  3057  3081  3101  3162  3180  3184  3194  3246  3507
        3674  4247  4308  4310  4332  4360  4364  4368  4377  4457  4519  4521  4539  4541  4565
        4567  4569  4635  4685
COM     2121  2707  2726
DEC     1849  2335  2339  2388  2551  2598  2682  2719  2723  3141  3169  3343  3351  3465  3509
        3590  3 36  3924  3946  3950  4283
DECB    2785  2788  2863  2874
EMT     1078
HALT    1048  2656  2761
INC     1739  1799  1848  1862  1888  1899  1906  2137  2157  2331  2450  2492  2522  2537  2596
        2645  2869  2877  2921  3025  3032  3116  3143  3257  3339  3457  3537  3921  3949  3952
        4015  4073  4102  4320  4323  4334  4338  4341  4357  4370  4376  4385  4397  4578  4664
        4687
INCB    2639  2808  3262
IOT     1079
JMP     1066  1068  1070  1072  1786  1878  1969  1998  2033  2070  2106  2182  2223  2273  2314
        2364  2429  2512  2540  2617  3001  3084  3629  3633  3637  3644  3657  3679  3742  3804
        3835  3839  3843  3848  3855  3958  4080  4091  4096  4108  4199  4381  4384  4388  4391
        4401  4456  4461  4475  4484
JSR     1745  1764  1776  1777  1854  1983  2012  2014  2047  2052  2084  2087  2126  2129  2132
        2135  2146  2149  2152  2155  2198  2200  2242  2244  2246  2248  2250  2252  2299  2291
        2326  2329  2337  2346  2376  2381  2383  2386  2395  2396  2402  2403  2404  2441  2446
        2449  2489  2535  2612  2651  2780  2787  2794  2999  3060  3105  3424  3623  3625  3648
        3654  3672  3676  3682  3735  3737  3745  3829  3831  3851  3878  4045  4076  4103  4107
        4122  4129  4160  4167  4178  4225  4233  4276  4286  4317  4321  4335  4339  4440  4441
        4444  4464  4474  4478  4571  4602  4615  4624  4662  4672  4677
MOV     1689  1694  1697  1698  1702  1706  1708  1709  1710  1711  1712  1713  1717  1718  1721
        1722  1723  1724  1729  1731  1732  1733  1735  1755  1756  1757  1758  1759  1765  1766
        1772  1774  1775  1778  1781  1782  1783  1788  1789  1790  1791  1792  1793  1794  1795
        1798  1809  1811  1858  1872  1889  1901  1903  1912  1918  1923  1924  1970  1971  1972
        1973  1975  1976  1977  1982  1999  2000  2001  2002  2004  2005  2006  2010  2011  2013
        2034  2035  2036  2037  2039  2040  2041  2045  2046  2051  2071  2072  2073  2074  2076
        2077  2078  2082  2083  2085  2086  2107  2108  2109  2110  2112  2113  2114  2122  2123
        2125  2131  2144  2145  2151  2183  2184  2185  2186  2188  2189  2190  2194  2195  2196
        2197  2199  2224  2225  2226  2227  2229  2230  2231  2235  2236  2237  2241  2274  2275
        2276  2277  2279  2280  2281  2285  2286  2287  2288  2290  2295  2315  2316  2317  2318
        2320  2321  2322  2325  2327  2328  2330  2334  2336  2338  2345  2365  2366  2367  2368
        2370  2371  2372  2373  2375  2377  2378  2379  2380  2382  2384  2385  2387  2394  2401
        2430  2431  2432  2433  2435  2436  2437  2440  2442  2445  2447  2448  2480  2486  2488
        2498  2518  2519  2524  2525  2534  2545  2546  2558  2583  2591  2601  2609  2641  2646
        2660  2663  2687  2695  2696  2701  2704  2705  2713  2717  2729  2763  2764  2768  2783
        2838  2846  2847  2848  2854  2861  2879  2880  2881  2882  2883  2904  2905  2906  2907
        2908  2909  2910  2915  2918  2938  2944  2945  2946  2947  2948  2950  2951  2979  2980
        2981  2982  2984  3023  3035  3066  3083  3098  3103  3132  3133  3136  3146  3155  3157
        3168  3199  3200  3201  3202  3235  3236  3238  3241  3250  3260  3261  3263  3264  3267
        3268  3290  3291  3292  3293  3294  3295  3296  3297  3298  3299  3306  3307  3308  3309
```

DZRPZB.CMB CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

	3310	3311	3312	3313	3314	3315	3333	3334	3335	3337	3341	3345	3359	3360	3361
	3362	3363	3374	3375	3379	3385	3386	3422	3423	3426	3444	3445	3446	3447	3448
	3449	3450	3451	3503	3504	3510	3513	3514	3529	3530	3531	3532	3533	3534	3545
	3546	3547	3548	3549	3563	3565	3571	3572	3573	3574	3607	3608	3609	3610	3611
	3612	3614	3615	3616	3617	3618	3621	3622	3641	3642	3643	3646	3647	3651	3661
	3662	3684	3685	3702	3705	3706	3707	3708	3720	3721	3725	3728	3729	3730	3731
	3733	3734	3739	3740	3741	3744	3746	3750	3751	3762	3765	3777	3778	3779	3780
	3781	3782	3783	3784	3785	3786	3787	3788	3789	3800	3816	3817	3818	3819	3820
	3821	3822	3825	3827	3828	3845	3846	3847	3849	3856	3873	3875	3879	3884	3905
	3907	3908	3909	3910	3911	3912	3913	3914	3918	3919	3920	3928	3929	3930	3931
	3937	3938	3939	3943	3945	3954	3955	3956	3957	3987	3992	3999	4007	4038	4039
	4040	4041	4042	4044	4047	4050	4056	4057	4058	4059	4061	4063	4068	4082	4083
	4085	4086	4093	4094	4095	4097	4083	4112	4113	4120	4121	4123	4124	4127	4128
	4130	4131	4137	4138	4139	4157	4158	4159	4164	4165	4166	4171	4172	4173	4182
	4184	4186	4187	4188	4190	4192	4218	4221	4224	4240	4253	4275	4285	4307	4319
	4331	4337	4402	4410	4416	4417	4418	4425	4427	4435	4439	4443	4463	4470	4472
	4476	4489	4490	4491	4492	4493	4498	4579	4596	4597	4598	4610	4641	4643	4644
	4645	4660	4675	4693	4694										
MOVB	1716	1753	1773	1913	1978	1981	2007	2009	2042	2044	2079	2081	2115	2117	2191
	2193	2232	2234	2282	2284	2323	2438	2648	2679	2681	2709	2710	2765	2793	2801
	2839	2840	2843	2844	2845	2849	2852	2853	2872	2913	2916	2930	2933	2942	2994
	3019	3031	3047	3062	3074	3142	3161	3166	3172	3177	3192	3266	3377	3464	3575
	3578	3586	3613	3665	3722	3723	3724	3726	3823	4114	4523	4543			
NEG	2850	2912	3340	3344	3355	3356									
NOP	2613	2614	2615												
RESET	1699	1763	2611												
ROL	2856	2858	2859	2860	2862	3536									
ROR	2711	3349	3350												
RTI	1730	2665	2770	2884	2952	3024	3036	3104	3137	3147	3203	3269	3300	3316	3387
RTS	2559	2738	2810	2985	3364	3380	3427	3470	3515	3550	3593	3687	3767	3790	3858
	3890	4024	4140	4191	4258	4299	4507	4526	4546	4580	4646	4695			
SBC	3357	3454													
SUB	2053	2161	2202	2238	2245	2249	2547	2647	2919	3453	3455	3882			
SWAB	4177	4254													
TRAP	3389	3398	3399	3400	3401	3403	3405	3406	3407	3408	3409				
TST	1747	1779	1784	1796	1852	1879	1881	1893	2127	2133	2147	2153	2577	2654	2661
	2733	2767	2775	2797	2867	2924	2934	2983	3000	3011	3016	3059	3096	3111	3139
	3164	3175	3198	3237	3358	3376	3885	3922	3923	3926	3942	3944	4001	4014	4017
	4018	4020	4154	4189	4195	4238	4239	4251	4252	4353	4362	4373	4382	4389	4423
	4432	4473	4477	4479	4501	4502	4525	4545	4573	4574	4575	4576	4577	4622	4623
	4631	4640	4642	4684	4688	4689									
TSTB	1816	1859	2759	2799	2926	2940	3017	3045	3071	3244	3505	4242	4245	4249	4293
	4312	4371	4379	4386	4451	4454	4459	4563	4632	4690					
.ASCII	1294	1295	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	4713		
.ASCIZ	1293	1296	1551	1744	1807	1822	1828	1834	1840	1846	1859	1877	1898	1917	2344
	2393	2400	2460	2464	2468	2485	2533	2557	2576	2582	2590	2607	3207	3208	3209
	3210	3212	3968	3972	3976	3980	3998	4012	4273	4305	4409	4700	4701	4702	4703
	4704	4705	4706	4708	4710	4722	4736	4743	4746	4749	4755	4763	4768	4776	4782
	4789	4796	4802	4809	4812	4819	4830	4833	4841	4850	4861	4870	4875	4884	
.BLKB	2966	3206	3491												
.BLKW	2957	4955	4956	4957	4958	4959	4960								
.BYTE	1251	1252	1257	1258	1266	1267	1275	1276	1277	1278	1332	1333	1334	1335	1336
	1337	1338	1339	1343	1344	1345	1346	1347	1348	1349	1350	1373	1374	1376	1377
	1814	1815	1891	1892	1920	1921	2585	2596	2619	2885	2886	2887	2888	3204	3205
	4413	4414	4920	4921	4924	4925	4928	4929	4932	4933	4936	4937	4939	4940	4943
	4944	4946	4947												

```
.DSABL  3121
.ENABL     1  1013  2962
.END    4963
.ENDC   1018  1031  1033  1034  1051  1055  1059  1061  1078  1170  1184  1245  1249  1251  1279
        1290  1291  1292  1293  1294  1298  1554  1689  1706  1707  1710  1712  1714  1715  1717
        1719  1735  1744  1749  1755  1807  1815  1816  1822  1828  1834  1640  1846  1869  1877
        1892  1893  1898  1917  1921  1922  1932  1953  1957  1958  1965  1966  1970  1972  1978
        1987  1988  1994  1995  1999  2001  2007  2018  2019  2029  2030  2034  2036  2042  2059
        2060  2066  2067  2071  2073  2079  2094  2095  2102  2103  2107  2109  2115  2167  2168
        2178  2179  2183  2185  2191  2210  2211  2219  2220  2224  2226  2232  2258  2259  2269
        2270  2274  2276  2282  2302  2303  2310  2311  2315  2317  2323  2344  2351  2352  2360
        2361  2365  2367  2373  2393  2400  2410  2411  2425  2436  2430  2432  2438  2460  2464
        2468  2485  2533  2557  2567  2569  2571  2576  2592  2586  2587  2590  2594  2600  2603
        2604  2607  2609  2611  2617  2619  2620  2626  2629  2639  2646  2651  2652  2653  2654
        2665  2666  2668  2745  2765  2816  2894  2962  2993  3002  3006  3037  3039  3054  3055
        3121  3125  3136  3148  3149  3157  3159  3162  3190  3207  3208  3214  3218  3221  3225
        3230  3232  3243  3244  3246  3248  3255  3257  3262  3263  3267  3270  3271  3275  3321
        3369  3375  3378  3397  3398  3399  3400  3401  3402  3403  3404  3405  3406  3407  3408
        3409  3410  3414  3433  3496  3520  3555  3596  3690  3770  3793  3807  3861  3894  3968
        3972  3976  3980  3998  4012  4027  4086  4089  4124  4125  4131  4132  4134  4136  4143
        4202  4261  4273  4305  4409  4414  4415  4509  4529  4549  4583  4649
.EQUIV  1078  1079  1087  1102  1103  1132  1133  1134  1135  1136  1137  1138  1139  1140  1141
        1160  1161  1162  1163  1164  1165  1166  1167  1168  1169
.EVEN   1326  1553  1744  1807  1822  1828  1834  1840  1846  1869  1877  1898  1917  2344  2393
        2400  2460  2464  2468  2485  2533  2557  2576  2582  2590  2607  2620  2968  3968  3972
        3976  3980  3998  4012  4273  4305  4409  4731  4894  4949
.IF     1014  1031  1032  1033  1034  1051  1054  1057  1059  1076  1142  1170  1244  1248  1250
        1279  1290  1291  1292  1293  1297  1298  1689  1701  1706  1708  1710  1712  1714  1715
        1717  1735  1743  1746  1749  1806  1814  1815  1821  1827  1833  1839  1845  1868  1876
        1891  1892  1897  1916  1920  1921  1929  1950  1956  1958  1965  1969  1971  1978  1986
        1988  1994  1998  2000  2007  2017  2019  2029  2033  2035  2042  2058  2060  2066  2070
        2072  2079  2093  2095  2102  2106  2108  2115  2166  2168  2178  2182  2184  2191  2209
        2211  2219  2223  2225  2232  2257  2259  2269  2273  2275  2292  2301  2303  2310  2314
        2316  2323  2343  2350  2352  2360  2364  2366  2373  2392  2399  2409  2411  2425  2429
        2431  2438  2459  2463  2467  2484  2532  2556  2566  2567  2568  2569  2570  2571  2573
        2575  2581  2585  2586  2589  2599  2602  2604  2606  2609  2611  2617  2619  2620  2625
        2628  2639  2642  2649  2651  2652  2654  2658  2665  2666  2667  2744  2765  2815  2893
        2961  2963  2991  2996  3002  3038  3039  3053  3077  3124  3125  3135  3148  3156  3158
        3162  3163  3206  3207  3214  3217  3220  3225  3230  3242  3244  3245  3246  3255  3257
        3263  3264  3269  3270  3271  3274  3320  3368  3374  3378  3389  3398  3399  3400  3401
        3402  3403  3405  3406  3407  3408  3409  3410  3413  3432  3495  3519  3554  3595  3689
        3769  3792  3806  3860  3893  3967  3971  3975  3979  3997  4011  4026  4085  4086  4123
        4125  4130  4132  4133  4135  4142  4201  4260  4272  4304  4408  4413  4414  4508  4528
        4548  4582  4648
.IFF    1031  1033  1034  1055  1059  1061  1076  1245  1248  1250  1279  1298  1706  1815  1893
        1921  1956  1957  1958  1966  1970  1972  1978  1986  1987  1988  1995  1999  2001  2007
        2017  2018  2019  2030  2034  2036  2042  2058  2059  2060  2067  2071  2073  2079  2093
        2094  2095  2103  2107  2109  2115  2166  2167  2168  2179  2183  2185  2191  2209  2210
        2211  2220  2224  2226  2232  2257  2258  2259  2270  2274  2276  2282  2301  2302  2303
        2311  2315  2317  2323  2350  2351  2352  2361  2365  2367  2373  2409  2410  2411  2426
        2429  2432  2438  2567  2570  2573  2586  2599  2602  2619  2626  2628  2642  2665  2666
        2668  2745  2816  2894  2962  3039  3054  3064  3125  3128  3136  3148  3149  3158  3190
        3206  3218  3243  3244  3246  3270  3275  3321  3369  3375  3414  3433  3496  3520  3555
        3696  3690  3770  3793  3807  3861  3894  4027  4085  4089  4125  4132  4134  4136  4143
        4202  4261  4414  4509  4529  4549  4583  4649
.IFT    1744  1807  1822  1828  1834  1840  1846  1869  1877  1898  1917  2344  2393  2400  2460
```

```
        2464  2468  2485  2533  2557  2576  2582  2590  2607  2652  3127  3132  3254  3968  3972
        3976  3980  3998  4012  4125  4132  4273  4305  4409
.IFTF   1744  1807  1822  1828  1834  1840  1846  1869  1877  1898  1917  2344  2393  2400  2460
        2464  2468  2485  2533  2557  2576  2582  2590  2607  2651  3064  3125  3128  3252  3968
        3972  3976  3980  3998  4012  4124  4131  4273  4305  4409
.IIF    1013  1018  1023  1028  1029  1030  1031  1034  1035  1036  1037  1038  1039  1048  1297
        1707  1710  1714  1715  1717  1718  1812  1890  1919  2569  2584  2592  2594  2595  2619
        2620  2629  2630  2631  2632  2633  2638  2657  2665  2666  2714  2718  2812  2962  2968
        3007  3008  3067  3198  3207  3214  3221  3222  3223  3224  3225  3229  3253  3254  3267
        3270  3271  3397  3398  3399  3400  3401  3403  3405  3406  3407  3408  3409  4222  4411
.IRP    1298  1689  1787  1956  1986  2017  2058  2093  2166  2209  2257  2301  2350  2409  2573
        2904  2944  3290  3310  3333  3361  3529  3547  3779  3785
.LIST      1  1013  1048  1184  1279  1281  1282  1283  1284  1285  1286  1287  1288  1289  1290
        1291  1685  1689  1719  1744  1807  1822  1828  1834  1840  1846  1869  1877  1898  1917
        1928  1954  1956  1978  1986  2007  2017  2042  2058  2079  2093  2115  2166  2191  2209
        2232  2257  2282  2301  2323  2344  2350  2373  2393  2400  2409  2438  2460  2464  2468
        2485  2533  2557  2576  2582  2590  2594  2607  2611  2623  2665  3148  3225  3389  3397
        3398  3399  3400  3401  3402  3403  3404  3405  3406  3407  3408  3409  3410  3968  3972
        3976  3980  3998  4012  4123  4130  4273  4305  4409
.MACRO     1  1034  1242  1685  1956  1986  2017  2058  2093  2166  2209  2257  2301  2350  2409
        3389
.MCALL  1013  1184  1719
.NLIST     1  1013  1048  1184  1279  1281  1282  1283  1284  1285  1286  1287  1288  1289  1290
        1291  1685  1689  1719  1744  1807  1822  1828  1834  1840  1846  1869  1877  1898  1917
        1928  1954  1956  1978  1986  2007  2017  2042  2058  2079  2093  2115  2166  2191  2209
        2232  2257  2282  2301  2323  2344  2350  2373  2393  2400  2409  2438  2460  2464  2468
        2485  2533  2557  2576  2582  2590  2594  2607  2611  2623  2665  3148  3225  3389  3397
        3398  3399  3400  3401  3402  3403  3404  3405  3406  3407  3408  3409  3410  3968  3972
        3976  3980  3998  4012  4123  4130  4273  4305  4409
.NTYPE  4123  4130
.PAGE   1242  1554  1685  1925
.REM       1
.REPT   1048  1281  1290  1929  1950  3710  3753  4116
.SBTTL  1024  1042  1052  1063  1074  1242  1554  1686  1700  1746  1926  1956  1986  2017  2058
        2093  2166  2209  2257  2301  2350  2409  2564  2622  2623  2668  2742  2813  2891  2959
        3215  3272  3318  3366  3389  3411  3430  3493  3517  3555  3596  3690  3770  3793  3807
        3961  3894  4027  4143  4202  4261  4509  4529  4549  4583  4649  4698  4734  4828  4896
        4917
.TITLE  1013
.WORD   1048  1049  1050  1060  1250  1253  1254  1255  1256  1259  1260  1261  1262  1263  1264
        1265  1268  1269  1270  1279  1281  1282  1283  1284  1285  1286  1287  1288  1289  1290
        1298  1299  1300  1301  1302  1303  1304  1305  1307  1308  1309  1310  1311  1312  1313
        1314  1315  1316  1319  1320  1321  1322  1323  1324  1325  1353  1354  1355  1356  1357
        1358  1359  1360  1361  1362  1363  1364  1365  1366  1367  1368  1375  1380  1381  1382
        1383  1384  1385  1388  1389  1390  1391  1392  1393  1394  1395  1396  1397  1398  1401
        1402  1403  1404  1405  1408  1409  1410  1411  1412  1415  1416  1418  1420  1422  1424
        1426  1428  1430  1431  1433  1438  1439  1440  1443  1444  1445  1446  1447  1448  1451
        1452  1453  1454  1455  1456  1459  1460  1461  1462  1463  1464  1467  1468  1469  1470
        1471  1472  1475  1476  1477  1478  1479  1480  1483  1484  1485  1486  1487  1488  1491
        1492  1493  1494  1495  1496  1499  1500  1503  1504  1505  1506  1509  1510  1514  1515
        1516  1517  1518  1519  1520  1521  1522  1523  2130  2136  2150  2156  2347  2405  2599
        2602  2618  2691  2700  2731  2809  2889  2963  2964  2965  3396  3428  3466  3512  3551
        3552  3624  3665  3736  3830  3960  4046  4220  4395  4437  4898  4899  4901  4904  4907
        4909  4911  4914  4919  4923  4927  4931  4935  4938  4942  4945  4951  4952  4953
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,NOW.SEQ/SOL/CRF/PAGNUM/NL:TOC=DZRFZB.SML,DZRPZB.CMB
RUN-TIME: 44 60 8 SECONDS
RUN-TIME RATIO: 318/114=2.7
CORE USED:  36K  (71 PAGES)
```

K10

Spooler runtime 18 Seconds, 7k KCS, 511 disk reads, 3 disk writes, 12k pages.