RK611/RK06

DRIVE COMPATIBILITY PROG

MD-11-DZR6Q-A

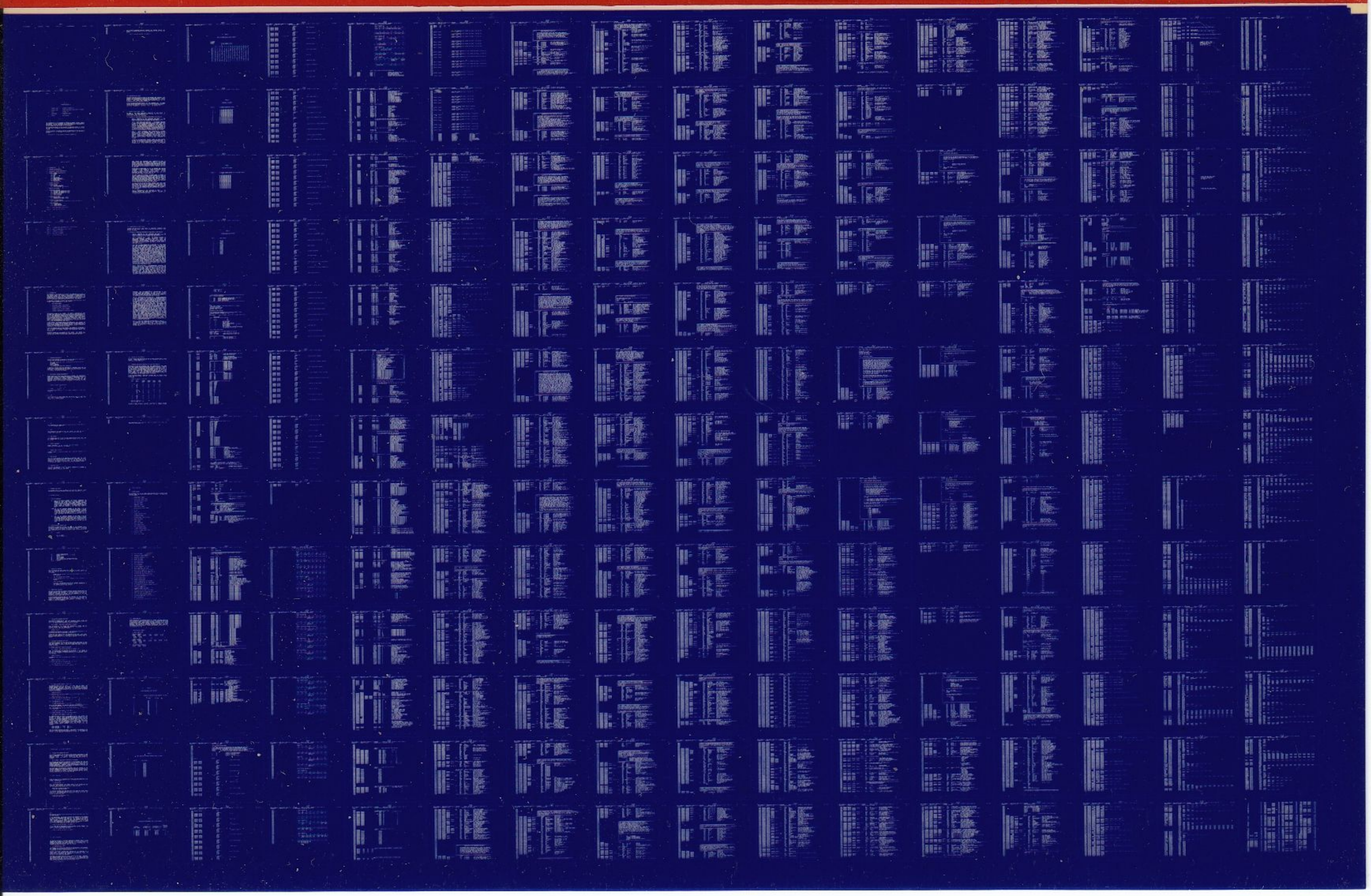
EP-DZR6Q-A-DL-A

FICHE 1 OF 2

JUN 1977

digital

MADE IN USA



RK611/RK06

DRIVE COMPATIBILITY PROG

MD-11-DZR6Q-A

EP-DZR6Q-A-DL-A

FICHE 2 OF 2

JUN 1977

digital

MADE IN USA

B01

EOF1DZLAEBSEQ	00010000	770526	PDP10 411	N-HDR1DZR6QASEQ	00010000	770526

```
                                .REM @
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZR6Q-A-D
PRODUCT NAME:	UNIBUS RK06 DRIVE COMPATIBILITY PROGRAM
DATE:	JANUARY 1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DAVE HOFFMAN

TABLE OF CONTENTS

TABLE C - CYLINDER BLOCK ASSIGNMENT FOR A GIVEN SURFACE

TABLE D - BASIC CYLINDER BLOCK LAYOUT EXAMPLE

TABLE E - OVERWRITE CYLINDERS

TABLE F - SELF-TEST CYLINDERS

TABLE G - PSEUDO-RANDOM DATA PATTERN

1.0 INTRODUCTION

THE PURPOSE OF THIS PROGRAM IS TO VERIFY THE COMPATIBILITY OF UP TO 16 RK06 DRIVES WHICH MAY RESIDE ON ONE OR MORE RK611/RK06 SUBSYSTEMS (INCLUDING SYSTEMS WITH MULTIPLE SUBSYSTEMS). COMPATIBILITY IS DEFINED HERE AS THE ABILITY OF A DRIVE TO WRITE DATA WHICH CAN BE READ SUCCESSFULLY BY ALL OTHER DRIVES, AND ADDITIONALLY THE ABILITY OF A DRIVE TO COMPLETELY OVER-WRITE DATA WRITTEN BY ALL OTHER DRIVES.

THE PROGRAM IS DESIGNED TO DETECT THE FOLLOWING CONDITIONS WHICH MOST COMMONLY CAUSE INCOMPATIBILITY BETWEEN DRIVES:

1. HEAD MIS-ALIGNMENT
2. POSITIONER LATERAL MISALIGNMENT
3. SPINDLE-CARTRIDGE INTERFACE RUNOUT
4. IMPROPER LEVELS OF WRITE CURRENT
5. INCORRECT ADDRESSING OF READ/WRITE HEADS

THE TESTING IS DONE IN TWO PASSES. IN PASS 1, COMPATIBILITY DATA PATTERNS ARE WRITTEN BY ALL THE DRIVES UPON THE SAME DISK CARTRIDGE, AND THE BASIC READ/WRITE CAPABILITY OF EACH DRIVE IS DEMONSTRATED. IN PASS 2, THE COMPATIBILITY DATA FROM ALL DRIVES IS READ BY EACH DRIVE, WITH INCREASING HEAD OFFSET, AND THIS IS COMPARED WITH EACH DRIVE'S ABILITY TO READ ITS OWN DATA. IN ADDITION, EACH DRIVE'S CAPABILITY TO OVERWRITE DATA WRITTEN BY ALL OTHER DRIVES IS TESTED ON THE SECOND PASS. (FOR THE REMAINDER OF THIS SPECIFICATION, THE ABOVE DEFINITIONS OF THE FIRST AND SECOND PASS SHALL APPLY).

IN BOTH PASSES, THE PROGRAM DIRECTS THE OPERATOR IN THE LOADING AND UNLOADING OF DRIVES AND THE MOVEMENT OF THE CARTRIDGE FROM DRIVE TO DRIVE, THROUGH MESSAGES AT THE CONSOLE TERMINAL. AT THE COMPLETION OF TESTING ON EACH DRIVE DURING THE SECOND PASS A SUMMARY IS PRINTED OF COMPATIBILITY TEST RESULTS FOR THAT DRIVE.

WITHIN THE VARIOUS TESTS OF BOTH PASSES, THE CAPABILITY IS PROVIDED TO LOOP ON CURRENT OPERATIONS, AND SWITCH REGISTER OPTIONS ARE PROVIDED, FOR A VARIETY OF LOOPING, RUNNING, AND REPORTING MODES (SEE SECTION 6.2).

UNEXPECTED ERRORS WILL BE REPORTED AS THEY OCCUR. THE REPORT WILL INCLUDE A TEST NUMBER AND DESCRIPTION, CURRENT AND PREVIOUS OPERATIONS GOOD AND BAD TEST DATA, AND APPLICABLE DEVICE REGISTER CONTENTS.

2.0 HARDWARE REQUIREMENTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN THE RK06 DRIVE COMPATIBILITY PROGRAM, FOR EACH SUBSYSTEM WHICH MAY BE PRESENT:

PDP-11/04, (05,10 MFG. ONLY), 20,30,34,35,40,45,50,70, OR PDQ
16K MEMORY
CONSOLE TERMINAL
RK611 CONTROLLER
1 TO 8 RK06 DISK DRIVES PER CONTROLLER

IN ADDITION, A SINGLE RK06 DISK CARTRIDGE IS REQUIRED WHICH MAY BE FORMATTED IN EITHER 20 OR 22 SECTOR FORMAT, ON A RELIABLE WELL-ALIGNED RK06 DRIVE. THIS CARTRIDGE WILL BE MOVED FROM DRIVE TO DRIVE, (ON UP TO 16 DRIVES) ON EACH OF TWO PASSES.

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE RK06 DRIVE COMPATIBILITY PROGRAM, THE SUBSYSTEM(S) UNDER TEST SHOULD BE CAPABLE OF PASSING THE CONTROLLER DIAGNOSTICS DZR6A-DZR6E AND DZR6K, THE DRIVE DIAGNOSTICS DZR6H-DZR6J, AND THE SUBSYSTEM VERIFICATION PROGRAMS DZR6M-DZR6N. IN ADDITION, THE CARTRIDGE MUST BE FORMATTED IN 20 OR 22 SECTOR FORMAT USING THE PACK FORMATTER DZR6L, OR EQUIVALENT (PERFORMED ON KNOWN GOOD, ALIGNED DRIVE).

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 SYSMAC

THIS PROGRAM USES PORTIONS OF THE SYSMAC DIAGNOSTIC SYSTEM MACRO PACKAGE.

4.2 XXDP

THIS PROGRAM MAY BE LOADED UNDER XXDP, AND MAY BE RUN IN DUMP MODE ONLY. DUE TO MANUAL INTERVENTION AND LACK OF END-OF-PASS HOOKS, THE PROGRAM IS NOT XXDP CHAINABLE.

4.3 ACT

THIS PROGRAM MAY BE LOADED UNDER ACT AND MAY BE RUN IN DUMP MODE ONLY.
IT IS NOT CHAINABLE UNDER ACT.

4.4 APT

THIS PROGRAM MAY BE LOADED BY THE APT SYSTEM, BUT MAY BE RUN IN PROGRAM (DUMP) MODE ONLY. IT CANNOT BE RUN IN APT SCRIPT MODE.

4.5 DUAL-ACCESS

THIS PROGRAM DOES NOT UTILIZE THE DUAL-ACCESS OPTION IN ANY WAY, AND ALL DRIVES UNDER TEST SHOULD BE DE-SELECTED THROUGH THE PORT WHICH IS NOT IN USE.

4.6 MEMORY MANAGEMENT

MEMORY MANAGEMENT IS NOT UTILIZED IN THIS PROGRAM. IF IT IS INSTALLED, IT IS DISABLED BY THE PROGRAM.

4.7 MEMORY PARITY OPTION

IF PARITY MEMORY IS INSTALLED, MEMORY PARITY TRAPS ARE DISABLED BY THE PROGRAM.

4.8 BAD SECTORS

THE LIST OF BAD SECTORS ON THE CARTRIDGE IS OBTAINED FROM THE FIRST DRIVE TO BE TESTED ON THE CURRENT SUBSYSTEM. ACCORDING TO A SWITCH REGISTER OPTION (SEE SECTION 6.2) THIS LIST MAY BE TYPED AT THE CONSOLE AT THE START OF THE FIRST PASS. ALL DATA ERRORS OCCURING IN THE PROGRAM ARE IGNORED IF THEY OCCUR IN SECTORS DESIGNATED AS BAD.

4.9 EXECUTION TIME

THE TOTAL TIME REQUIRED TO RUN THE DRIVE COMPATIBILITY PROGRAM IS DIRECTLY PROPORTIONAL TO THE NUMBER OF DRIVES TO BE TESTED AND REQUIRES ABOUT 8-10 MINUTES PER DRIVE.

5.0 PROGRAM LOAD MEDIA

THIS PROGRAM CAN BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR FROM THE ACT OR APT SYSTEMS OR FROM ANY MEDIA SUPPORTED BY XXDP.

6.0 PROGRAM OPTIONS

6.1 STARTING ADDRESSES

200 - THIS IS THE STARTING ADDRESS FOR DEFAULT PARAMETERS AND RUNNING OF PASS 1 AND PASS 2 ON A SINGLE SUBSYSTEM. THE PROGRAM WILL USE DEFAULT RK611 BASE ADDRESS, INTERRUPT VECTOR AND PRIORITY, AND WILL AUTOMATICALLY DETERMINE WHICH DRIVES TO TEST. DRIVES MUST BE ON-LINE, POWERED UP, AND EITHER LOADED OR UNLOADED. THE PROGRAM WILL ASSUME ALL DRIVES TO BE TESTED RESIDE ON ONE RK611/RK06 SUBSYSTEM ONLY.

204 - THIS IS THE STARTING ADDRESS TO RUN PASS 1 ON ALL RK611/RK06 SUBSYSTEMS WHICH RESIDE ON THIS PDP-11 SYSTEM. THE PROGRAM WILL ASK FOR THE RK611 BASE ADDRESS, INTERRUPT VECTOR, AND PRIORITY FOR EACH SUBSYSTEM ON THIS SYSTEM, AND IT ASKS FOR THE LETTER NAMES (A THRU P) ASSIGNED TO ALL OTHER SUBSYSTEMS, AND THE DRIVE(S) WHICH WILL BE TESTED ON EACH.

220 - THIS IS THE STARTING ADDRESS TO RUN PASS 2 ON ALL RK611/RK06 SUBSYSTEMS WHICH RESIDE ON THIS PDP-11 SYSTEM. THE PROGRAM WILL ASK FOR THE RK611 BASE ADDRESS, INTERRUPT VECTOR, AND PRIORITY FOR EACH SUBSYSTEM ON THIS SYSTEM, AND IT ASKS FOR THE LETTER NAMES (A THRU P) ASSIGNED TO ALL OTHER SUBSYSTEMS, AND THE DRIVE(S) WHICH WILL BE TESTED ON EACH.

6.2 SWITCH REGISTER OPTIONS USED

THIS PROGRAM IS DESIGNED TO ALLOW THE USE OF THE HARDWARE SWITCH REGISTER IF PRESENT, OR THE SYSMAC-SUPPORTED SOFTWARE SWITCH REGISTER (IF HARDWARE SWR IS NOT PRESENT, OR IS SET TO ALL ONES). IN EITHER CASE, THE FOLLOWING OPTIONS ARE IMPLEMENTED WHEN THE APPROPRIATE BITS ARE SET TO 1:

BIT	OPTION
15	HALT ON ERROR
14	LOOP ON CURRENT TEST

13	INHIBIT ERROR REPORTS
12	REPORT DESCRIPTION ONLY, ON ERRORS
11	UNUSED
10	BELL ON ERROR
09	LOOP ON ERROR
08	APPLY RANDOM STALL BETWEEN OPERATIONS
07	TYPE BAD SECTOR FILES (BSF'S) AT START
06-00	UNUSED

7.0 RUNNING THE PROGRAM

ONCE THE PROGRAM HAS BEEN LOADED INTO CORE (IN A GIVEN SYSTEM, IF THERE ARE MULTIPLE SYSTEMS) THE FOLLOWING STEPS MUST BE TAKEN TO RUN THE PROGRAM:

1. INSURE THAT ALL DRIVES TO BE TESTED ARE POWERED UP AND SINGLE PORT SELECTED.
2. LOAD THE DESIRED START ADDRESS.
3. SET ANY DESIRED BITS IN THE HARDWARE SWITCH REGISTER (IF PRESENT).
4. START THE PROGRAM.
5. FOLLOW ALL INSTRUCTIONS TYPED BY THE PROGRAM PERTAINING TO THE MANUAL INTERVENTION REQUIRED, AND THE ALTERNATE USE OF MULTIPLE SYSTEMS (IF THERE ARE ANY).

8.0 OPERATIONAL DIALOGUE

THIS SECTION DESCRIBES THE CONSOLE TERMINAL DIALOGUE THROUGH WHICH THE PROGRAM DIRECTS THE OPERATOR, IN THE SELECTION OF OPTIONS AND THE LOADING AND UNLOADING OF DRIVES, AND THE MOVEMENT OF THE TEST CARTRIDGE. THE EXACT DIALOGUE WHICH IS USED DEPENDS UPON THE STARTING ADDRESS WHICH WAS CHOSEN (SEE SECTION 6.1).

IN THE FOLLOWING DISCUSSION AND IN THE PRINTOUT OF TEST RESULTS, DRIVES TO BE TESTED WILL BE REFERRED TO BY A LETTER AND A NUMBER. THE LETTER IS THE SUBSYSTEM LETTER NAME (OPERATOR ASSIGNED), AND THE NUMBER IS THE DRIVE NUMBER ON THAT SUBSYSTEM. FOR EXAMPLE, DRIVE C6 REFERS TO DRIVE 6 ON SUBSYSTEM C.

8.1 DIALOGUE FOR ADDRESS 200 START

THIS STARTING ADDRESS MAY BE USED FOR AUTOMATIC DRIVE SIZING AND DEFAULTING OF PARAMETERS, WHEN THE DRIVES RESIDE ON ONE SUBSYSTEM ONLY. THE PROGRAM FIRST IDENTIFIES ITSELF AS FOLLOWS:

MD-11-DZR6Q - RKO6 DRIVE COMPATIBILITY PROGRAM

THEN, THE DRIVES ON THE SYSTEM ARE AUTOMATICALLY SIZED, AND ALL EXISTING DRIVES WILL BE MARKED FOR TESTING. THE PROGRAM TYPES THE DRIVE LIST, AS IN THE FOLLOWING EXAMPLE:

DRIVES = 2,5,7

THE PROGRAM NOW PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE MOUNTING OF THE PACK, AS DESCRIBED IN SECTION 8.4.

PLEASE NOTE THAT THERE IS ONLY ONE SUBSYSTEM ON AN ADR. 200 START, AND IT IS NAMED SUBSYSTEM A. THE DRIVES IN THE ABOVE EXAMPLE WOULD BE REFERRED TO AS A2,A5,A7 IN THE TEST RESULTS PRINTOUT AT THE END OF PASS 2.

8.2 DIALOGUE FOR ADDRESS 204 START

THIS STARTING ADDRESS MUST BE USED ON EACH SYSTEM, WHEN THERE IS MORE THAN ONE SUBSYSTEM, BUT IT MAY ALSO BE USED WHEN THERE IS JUST ONE SUBSYSTEM (TOTAL), TO SPECIFY DRIVES TO TEST AND NON-DEFAULT PARAMETER VALUES, FOR PASS 1.

THE PROGRAM IDENTIFIES ITSELF AS FOLLOWS:

MD-11-DZR6Q - RKO6 DRIVE COMPATIBILITY PROGRAM

THEN, THE PROGRAM ASKS THE OPERATOR FOR THE DRIVES TO BE TESTED ON EACH OF THE POSSIBLE SUBSYSTEMS (STARTING WITH A - THE NAMES RANGE FROM SUBSYS A TO SUBSYS P. THERE COULD BE UP TO 16 SUBSYSTEMS, WITH A DRIVE ON EACH) :

SUBSYS A DRIVE(S) =

THE OPERATOR THEN TYPES THE DESIRED DRIVE NUMBERS, AS IN THE FOLLOWING EXAMPLE:

SUBSYS A DRIVE(S) = 2,5,7

THE PROGRAM THEN VERIFIES THE DRIVE NOS. BY TYPING :

WILL TEST DRIVE(S) 2,5,7 ON SUBSYS A.

NEXT, THE PROGRAM ASKS THE FOLLOWING QUESTION:

```
                    IS THERE ANOTHER SUBSYS (Y OR N <CR>)?

THE OPERATOR TYPES Y<CR> OR  N<CR>.  (IF  JUST  <CR>  IS  TYPED,  THE
PROGRAM  ASSUMES  THAT N<CR> WAS TYPED).  IF THE OPERATOR TYPED N, THE
PROGRAM PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE MOUNTING
OF THE PACK, AS DESCRIBED IN SECTION 8.4.  IF Y WAS TYPED, THE PROGRAM
ASKS FOR THE NUMBERS OF THE DRIVES TO BE TESTED ON THE NEXT  SUBSYSTEM
(SUBSYS B) AS FOLLOWS:

     SUBSYS B DRIVE(S) =

THE OPERATOR TYPES THE DRIVE NUMBERS, AS IN THE FOLLOWING EXAMPLE:

     SUBSYS B DRIVE(S) = 2,3

THE PROGRAM THEN VERIFIES THE DRIVE NOS.  BY TYPING :

     WILL TEST DRIVE(S) 2,3 ON SUBSYS B.

NEXT, THE PROGRAM WILL ASK :

     IS THERE ANOTHER SUBSYS (Y OR N <CR>)?

AND IN THE SAME MANNER, THE OPERATOR SPECIFIES THE DRIVES ON  EACH  OF
THE REMAINING SUBSYSTEMS, UNTIL ALL HAVE BEEN SPECIFIED.

NEXT, THE PROGRAM ASKS FOR THE LETTER NAME ASSIGNED TO  THE  SUBSYSTEM
TO BE TESTED NEXT :

     TYPE NAME OF NEXT SUBSYS TO TEST (A-P) :

THE OPERATOR RESPONDS, AS IN THE FOLLOWING EXAMPLE :

     TYPE NAME OF NEXT SUBSYS TO TEST (A-P) :  A

ALL SUBSYSTEMS MUST BE TESTED IN THE ORDER IN WHICH  THE  LETTERS  ARE
ASSIGNED  (A  THRU P).  NEXT, THE PROGRAM ALLOWS THE OPERATOR TO ALTER
THE RK611 BUS ADDRESS, VECTOR ADDRESS, AND INTERRUPT PRIORITY FOR THIS
SUBSYSTEM.   FOR  EACH  PARAMETER  THE CURRENT VALUE IS TYPED, AND THE
OPERATOR IS GIVEN THE OPPORTUNITY TO TYPE IN A NEW VALUE,  PLUS  <CR>.
IF JUST <CR> IS TYPED, THE PARAMETER IS NOT CHANGED.  WHEN THE PROGRAM
IS FIRST LOADED, THE FOLLOWING DEFAULT VALUES ARE ASSIGNED:  RK611 BUS
ADDRESS = 177440,   VECTOR   ADDRESS = 210,   AND  PRIORITY = 5.   THE
FOLLOWING EXAMPLE SHOWS A  PRINTOUT  IN  WHICH  ONLY  THE  VECTOR  WAS
CHANGED:

     RK611 BUS ADR  =  177440    NEW =
     RK611 VEC ADR  =     210    NEW = 240
     RK611 PRIORITY =       5    NEW =

THEN THE PROGRAM PROCEEDS WITH PASS 1, AND DIRECTS THE OPERATOR IN THE
MOUNTING  OF THE PACK, AS DESCRIBED IN SECTION 8.4.  AT THE COMPLETION
OF PASS 1 ON THE SUBSYSTEM, THE PROGRAM WILL INFORM THE  OPERATOR  HOW
```

TO PERFORM PASS 1 ON THE NEXT SUBSYSTEM.

8.3 DIALOGUE FOR ADDRESS 220 START

THIS STARTING ADDRESS MUST BE USED ON EACH SYSTEM, WHEN THERE IS MORE THAN 1 SUBSYSTEM, BUT IT MAY ALSO BE USED WHEN THERE IS JUST ONE SUBSYSTEM (TOTAL), TO SPECIFY DRIVES TO TEST AND NON-DEFAULT PARAMETER VALUES, FOR PASS 2. THE PROGRAM IDENTIFIES ITSELF, AS FOLLOWS :

```
MD-11-DZR6Q - RK06 DRIVE COMPATIBILITY PROGRAM
```

THE DIALOGUE FOR 220 START IS IDENTICAL TO THE DIALOGUE FOR THE 204 START DESCRIBED ABOVE (SECTION 8.2), FOR THE SELECTION OF SUBSYSTEM PARAMETERS AND THE SPECIFICATION OF ALL DRIVES TO BE TESTED ON THE VARIOUS SUBSYSTEMS. HOWEVER, AFTER THIS DIALOGUE IS COMPLETED, THE PROGRAM PROCEEDS WITH PASS 2, AND DIRECTS THE OPERATOR IN THE MOVEMENT OF THE PACK, AS DESCRIBED IN SECTION 8.5.

NOTE THAT SINCE THE APPROPRIATE PROCESSOR MUST BE STARTED AT THE STARTING ADDRESS FOR EACH SUBSYSTEM TO BE TESTED, THE COMPATIBILITY TEST MAY BE PERFORMED IN STEPS, AT VARIOUS TIMES AND BETWEEN VARIOUS DISTANT LOCATIONS, BY MOVING THE TEST PACK AND SAVING THE PRINTOUT FROM EACH PASS ON EACH PDP-11 SYSTEM INVOLVED.

8.4 PASS 1 DIALOGUE

AFTER THE SELECTION OF PARAMETERS AND DRIVES HAS BEEN COMPLETED ON THE CURRENT SUBSYSTEM (SECTIONS 8.1-8.2), THE PROGRAM INDICATES THE START OF PASS 1 AS FOLLOWS:

```
** STARTING PASS 1 **
```

NEXT, THE PROGRAM SELECTS THE FIRST DRIVE TO BE TESTED ON THIS SUBSYSTEM, AND INSTRUCTS THE OPERATOR TO MOUNT THE TEST CARTRIDGE AND LOAD THE HEADS ON THAT DRIVE, AS IN THE FOLLOWING EXAMPLE:

```
MOUNT PACK ON DRIVE A2 AND LOAD.
TYPE R<CR> WHEN DRIVE READY:
```

THE OPERATOR PERFORMS THIS TASK AND TYPES R<CR> WHEN THE DRIVE READY LIGHT IS ON. THE PROGRAM PERFORMS PASS 1 FUNCTIONS ON THIS DRIVE (SEE SECTION 9.1) AND THEN INSTRUCTS THE OPERATOR TO UNLOAD THE DRIVE AND REMOVE THE PACK AS FOLLOWS:

```
UNLOAD DRIVE A2 AND REMOVE PACK.
TYPE R<CR> WHEN DONE:
```

THE OPERATOR PERFORMS THESE FUNCTIONS AND TYPES R<CR> AFTER THE PACK HAS BEEN REMOVED.

IN THE SAME MANNER, THE PROGRAM INSTRUCTS THE OPERATOR IN THE MOVEMENT OF THE PACK THROUGHOUT THE REST OF THE DRIVES ON THE CURRENT SUBSYSTEM. WHEN THIS HAS BEEN COMPLETED, THE PROGRAM DOES ONE OF THREE THINGS: (1) IF THERE IS ONLY ONE SUBSYSTEM (FROM ADR 200 START) THE PROGRAM BEGINS PASS 2 (SEE SECTION 8.5). (2) IF THERE IS ANOTHER SUBSYSTEM, THE PROGRAM DIRECTS THE OPERATOR TO PERFORM PASS 1 ON THE NEXT SUBSYS AS FOLLOWS (AND THEN HALTS) :

START AT ADR 204 FOR PASS 1 ON SUBSYS B

(3) IF THERE ARE NO MORE DRIVES TO TEST IN PASS 1 ON ANY SUBSYS, THE PROGRAM DIRECTS THE OPERATOR TO BEGIN PASS 2 ON THE FIRST SUBSYS (SEE SECT. 8.5) AS FOLLOWS (AND THEN HALTS) :

START AT ADR 220 FOR PASS 2 ON SUBSYS A

8.5 PASS 2 DIALOGUE

THE OPERATOR RETURNS TO THE FIRST SUBSYSTEM TO PERFORM PASS 2. EITHER THROUGH THE DIALOGUE OF THE ADR 200 START, OR AFTER THE SELECTION OF PARAMETERS AND DRIVES HAS BEEN COMPLETED IN ACCORDANCE WITH THE DIALOGUE OF THE ADR 220 START (SEE SECTION 8.3). IN EITHER CASE, THE PROGRAM INDICATES THE START OF PASS 2 BY TYPING :

** STARTING PASS 2 **

THE PROGRAM THEN DIRECTS THE OPERATOR IN THE UNLOADING, PACK MOVEMENT, AND LOADING OF ALL DRIVES ON THE FIRST SUBSYSTEM, IN THE SAME MANNER AS DESCRIBED FOR PASS 1 (SEE SECTION 8.4).

HOWEVER, AFTER PASS 2 TESTING (SEE SECTION 9.2) IS COMPLETED ON A GIVEN DRIVE, THE ENTIRE TEST RESULTS FOR THAT DRIVE ARE TYPED. THE DETAILS OF THIS PRINTOUT ARE DESCRIBED IN SECTION 10, AFTER THE DETAILS OF THE TESTING ARE DESCRIBED.

WHEN PASS 2 HAS BEEN COMPLETED FOR ALL DRIVES ON THE FIRST SUBSYSTEM, THE PROGRAM DOES ONE OF TWO THINGS: (1) IF THERE IS ONLY ONE SUBSYSTEM (FROM ADR 200 START) OR IF ALL DRIVES ON ALL SUBSYSTEMS HAVE BEEN TESTED IN PASS 2 (FROM ADR 220 START), THE ENTIRE TESTING AND REPORTING HAVE BEEN COMPLETED, AND THE PROGRAM TYPES:

** END OF TESTING **

```
602
603
604
605                     (2) IF THERE IS ANOTHER SUBSYSTEM, HOWEVER, THE  PROGRAM  DIRECTS  THE
606                     OPERATOR  TO PERFORM PASS 2 ON THE NEXT SUBSYSTEM AS FOLLOWS (AND THEN
607                     HALTS):
608
609                          START AT ADR 220 FOR PASS 2 ON SUBSYS B
```

9.0 DESCRIPTION OF TESTS

THE MAIN FUNCTIONAL BLOCKS OF CODE IN THE PROGRAM ARE ASSIGNED TEST NUMBERS, FOR THE PURPOSE OF IDENTIFICATION IN ERROR PRINTOUTS. TEST 0 REFERS TO THE OPERATOR INPUT DIALOGUE ROUTINES DESCRIBED IN SECTIONS 8.1-8.3. THE OTHER TEST NUMBERS ARE ASSIGNED BELOW, IN THE DESCRIPTION OF PASS 1 AND PASS 2 TESTING.

IN THE FOLLOWING SECTIONS, TABLES A-G ARE REFERRED TO. IN THESE TABLES, DRIVES ARE NAMED FROM 0-17 FOR ILLUSTRATIVE PURPOSES, ALTHOUGH THE DRIVES ARE NAMED THE FOLLOWING WAY IN AN ACTUAL SITUATION : A0,A1, A2,...B0,B1,B2,...C0,C1,C2,... ETC. (SEE SECTION 8.0).

9.1 DESCRIPTION OF PASS 1 TESTS

IN PASS 1, THE BASIC READ/WRITE CAPABILITY OF EACH DRIVE IS DEMONSTRATED, AND COMPATIBILITY DATA PATTERNS ARE WRITTEN BY ALL DRIVES UPON THE SAME TEST CARTRIDGE.

THE SEQUENCE OF OPERATIONS PERFORMED ON EACH DRIVE IS AS FOLLOWS:

1. TEST 1 - MOUNTING OF TEST CARTRIDGE FOR PASS 1 - THE OPERATOR MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.4).

2. TEST 2 - BASIC READ/WRITE DATA TEST - THE PROGRAM PERFORMS A WRITE AND WRITE CHECK OPERATION USING A "WORST CASE" DATA PATTERN, AT THE APPROPRIATE SECTOR FOR THIS DRIVE (SEE TABLE A) ON ALL SURFACES. THE PURPOSE OF THIS OPERATION IS TO VERIFY THE BASIC READ/WRITE CAPABILITY OF THE DRIVE ON PASS 1. THE ENTIRE SECTOR IS WRITTEN WITH THE REPETITION OF THE DATA PATTERN SHOWN IN TABLE B. THIS PATTERN CONSISTS OF A MIXTURE OF HIGH AND LOW FREQUENCY ELEMENTS AND BIT STRINGS REQUIRING VARIOUS DEGREES OF PRECOMPENSATION, WHEN ENCODED.

3. TEST 3 - WRITE OVERWRITE AND COMPATIBILITY CYLINDER BLOCKS - NEXT, THE PROGRAM WRITES ALL SECTORS FOR THIS DRIVE WITHIN THE CYLINDER BLOCKS SHOWN IN TABLE C ON ALL SURFACES USING A SINGLE REPEATED WORD OF THE PATTERN IN TABLE G. LOGICAL DRIVE 0 USES WORD 0, LOGICAL DRIVE 1 USES WORD 1, LOGICAL DRIVE 10 USES WORD 10, ETC. THUS, THE DATA FROM EACH DRIVE IS UNIQUE. TABLE C HAS BEEN DETERMINED AS FOLLOWS:

   IN EACH OF THE SEVEN WRITE CURRENT ZONES ON EACH SURFACE SECTORS ARE WRITTEN WITHIN TWO DISTINCT CYLINDER BLOCKS. IN THE FIRST 16 CYLINDERS OF EACH WRITE CURRENT ZONE DATA IS WRITTEN TO BE LATER OVERWRITTEN IN PASS 2, DURING THE OVERWRITE TEST. IN THE LAST 16 CYLINDERS OF EACH CURRENT

ZONE (EXCEPT THE INNERMOST ZONE) COMPATIBILITY DATA IS WRITTEN TO BE READ WITH OFFSET IN PASS 2. WITHIN EACH CURRENT ZONE BOTH THE OVERWRITE AND COMPATIBILITY CYLINDER BLOCKS ARE IDENTICALLY WRITTEN BUT FROM ZONE TO ZONE THE DATA WRITTEN BY EACH DRIVE IS ROTATED SEVERAL SECTORS SO THAT OVER ALL THE ZONES THE DATA APPEARS AT VARIOUS ANGULAR POSITIONS ON THE PACK.

WITHIN EACH CYLINDER BLOCK, UP TO 16 SECTORS ARE WRITTEN (DEPENDING ON THE NUMBER OF DRIVES BEING TESTED) ON EACH CYLINDER. THESE SECTORS ARE ALWAYS SECTORS 0,1,2,3,5,6,7,10,12,13,14,15,17,20,21,22(OCTAL). OF THE REMAINING SECTORS, 4,11,16, AND 23 ARE USED FOR DRIVE SELF-TESTING IN PASS 2 (SEE SECTION 9.2).

THE BASIC LAYOUT OF A TYPICAL CYLINDER BLOCK IS SHOWN IN TABLE D, WHERE THE BLOCK SHOWN IS THE COMPATIBILITY BLOCK FOR ZONE 1, WHICH STARTS ON CYLINDER 60, AND HAS THE ROTATING STARTING SECTOR = SECTOR 0. EACH NUMBER INSIDE THE BLOCK IS THE NUMBER OF THE DRIVE WHICH WRITES THAT SECTOR. TABLE D SHOWS THE BLOCKS WRITTEN BY EACH OF 16 DRIVES. IF ANY OF THE DRIVES SHOWN ARE NOT PRESENT, HOWEVER, THE BLOCKS RESERVED FOR THE MISSING DRIVES ARE SIMPLY NOT WRITTEN.

THE ABOVE PATTERN OF SECTOR WRITES INSURES THAT DATA FROM EACH DRIVE IS WRITTEN ON ADJACENT CYLINDERS TO DATA FROM EVERY OTHER DRIVE, IN BOTH DIRECTIONS. IN ADDITION, THE ROTATION OF THE ABOVE SECTORS FROM CURRENT ZONE TO CURRENT ZONE INSURES THAT OVERWRITE AND DATA COMPATIBILITY TESTING IS DONE AT SEVERAL DIFFERENT ANGULAR POSITIONS WITH RESPECT TO THE CARTRIDGE.

4. TEST 4 - DISMOUNTING OF TEST CARTRIDGE IN PASS 1 - THE OPERATOR UNLOADS THE DRIVE AND DISMOUNTS THE PACK, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.4), TO PROCEED WITH THE ABOVE STEPS ON THE NEXT DRIVE.

9.2 DESCRIPTION OF PASS 2 TESTS

IN PASS 2 THE ABILITY OF EACH DRIVE TO COMPLETELY OVERWRITE DATA WRITTEN BY ALL OTHER DRIVES AND TO READ DATA WRITTEN BY ALL OTHER DRIVES, IS TESTED.

THE SEQUENCE OF OPERATIONS PERFORMED BY EACH DRIVE IS AS FOLLOWS:

1. TEST 5 - MOUNTING OF TEST CARTRIDGE FOR PASS 2 - THE OPERATOR MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS, AS DIRECTED BY THE PROGRAM (SEE SECTION 8.5).

2. TEST 6 - OVERWRITE TEST - NEXT, THE PROGRAM PROCEEDS TO TEST THIS DRIVE'S OVERWRITE CAPABILITY. FIRST, THE APPROPRIATE CYLINDERS IN TABLE E FOR THIS DRIVE ARE OVERWRITTEN, ON EACH SURFACE. THE DATA USED IS A REPETITION OF A SINGLE WORD OF THE PATTERN IN TABLE G, LOGICAL DRIVE 0 USES WORD 0, LOGICAL DRIVE 1 USES WORD 1, LOGICAL DRIVE 10 USES WORD 10, ETC.

   THEN, EACH CYLINDER OVERWRITTEN IS READ BACK BY THIS DRIVE WITH THE FOLLOWING RANGE OF OFFSET VALUES : 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION (+ AND -). THE PROGRAM SCANS FOR READ ERRORS (DCK,HVRC,ETC.) DURING THIS READ, AND IF ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA HAS NOT BEEN CORRECTLY OVERWRITTEN, AND A SCORE FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS PERFORMING THE OVERWRITES. FOR EACH TRACK (0,1,2), SCORES ARE AVERAGED OVER ALL CYLS TESTED, IN EACH OFFSET DIRECTION. AT THE COMPLETION OF THE OVERWRITE TEST ON THIS DRIVE, THE SCORES OF ALL THE DRIVES ARE CONVERTED AND STORED, FOR PRINTING AT THE END OF PASS 2 (AS DESCRIBED IN SECTION 10.2). EACH SCORE IS PROPORTIONAL TO THE DEGREE OF OFFSET WHICH COULD BE APPLIED IN A GIVEN OFFSET DIRECTION BY THE CURRENT DRIVE WHILE SUCCESSFULLY READING THE DATA IT WROTE OVER ONE OF THE OTHER DRIVE'S DATA. THUS, THE PRINTOUT REVEALS WHICH DRIVES ARE INVOLVED, IN A SITUATION IN WHICH A DRIVE CANNOT OVERWRITE ONE OR SEVERAL OTHER DRIVE'S DATA.

3. TEST 7 - DRIVE SELF-TEST - THE PROGRAM NEXT EVALUATES THE DRIVE'S ABILITY TO WRITE AND READ ITS OWN DATA, AT VARIOUS POSITIONS ON THE PACK. FIRST, SECTORS 4,11,16, AND 23 OF THE APPROPRIATE CYLINDERS SHOWN IN TABLE F FOR THIS DRIVE ARE WRITTEN WITH THE DATA PATTERN SHOWN IN TABLE B, FOR ALL SURFACES. THEN, THE SECTORS ARE READ WITH THE FOLLOWING OFFSETS APPLIED : 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION. THE PROGRAM SCANS FOR READ ERRORS DURING EACH READ, AND IT COMPUTES A SCORE WHICH IS PROPORTIONAL TO THE FAILING OFFSET

MAGNITUDE. THEN, THE SCORES FOR ALL 4 SECTORS READ IN THIS CYLINDER BLOCK ARE AVERAGED, TO COME UP WITH A DRIVE SELF-TEST SCORE FOR EACH SURFACE FOR EACH OFFSET DIRECTION. THIS SCORE IS SAVED FOR LATER USE, TO BECOME THE STANDARD FOR THE COMPATIBILITY DATA READS WHICH ARE TO FOLLOW.

4. TEST 10 - COMPATIBILITY DATA READ TEST - HAVING ESTABLISHED A SELF-TEST SCORE FOR THIS DRIVE, THE PROGRAM PROCEEDS TO PERFORM THE COMPATIBILITY DATA READS OF THE PATTERNS WRITTEN BY ALL THE DRIVES IN EACH CYLINDER BLOCK (ON EACH SURFACE). EACH COMPATIBILITY CYLINDER BLOCK SHOWN IN TABLE C IS READ, A CYLINDER AT A TIME, FOR THE FOLLOWING OFFSET VALUES : 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200 MICRO-INCHES, IN EACH OFFSET DIRECTION. THE PROGRAM SCANS FOR READ ERRORS DURING EACH READ AND IF ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA WAS BEING READ AT THAT INSTANT AND A SCORE FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS READING THE COMPATIBILITY DATA. THEN, EACH SCORE IS APPROPRIATELY ADJUSTED TO REFLECT THE SELF-TEST SCORE FOR THE CURRENT DRIVE AT THAT PARTICULAR CYLINDER BLOCK. THE SCORES ARE THEN AVERAGED OVER ALL CYLINDER BLOCKS. EACH SCORE IS PROPORTIONAL TO THE CAPABILITY OF THE CURRENT DRIVE TO SUCCESSFULLY READ THE DATA WRITTEN BY ONE OF THE OTHER DRIVES, AND SCORES ARE COMPUTED SEPARATELY FOR EACH SURFACE (TRACK), FOR EACH OFFSET DIRECTION. THUS, THE PRINTOUT REVEALS WHICH DRIVES ARE INVOLVED IN A SITUATION IN WHICH A PARTICULAR DRIVE HAS DIFFICULTY IN READING THE DATA OF ONE OR SEVERAL OTHER DRIVES.

5. TEST 11 - TYPE TEST SCORES AND DISMOUNT PACK IN PASS 2 - THE OVERWRITE AND COMPATIBILITY DATA READ TEST SCORES FOR THIS DRIVE ARE CONVERTED AND TYPED. THEN, THE OPERATOR UNLOADS THE DRIVE AND DISMOUNTS THE PACK AS DIRECTED BY THE PROGRAM (SEE SECTION 8.5), TO PROCEED WITH THE ABOVE STEPS ON THE NEXT DRIVE.

10.0 PRINTOUT OF TEST RESULTS

THE TEST RESULTS ARE PRINTED AT THE END OF PASS 2 ON EACH DRIVE BEING TESTED. THESE RESULTS PERTAIN TO THE OVERWRITE TEST AND THE COMPATIBILITY DATA READ TEST.

10.1 OVERWRITE AND DRIVE COMPATIBILITY DATA TEST RESULTS

THE RESULTS OF BOTH THE OVERWRITE AND THE COMPATIBILITY DATA READ TESTS ARE ALWAYS PRINTED, REGARDLESS OF DEGREE OF SUCCESS. COMPLETE RESULTS ARE PRINTED, SEPARATELY FOR EACH DRIVE, AT THE END OF PASS 2 ON THAT DRIVE. THE TEST RESULTS ARE TABULAR IN FORM, AND CONSIST OF READ SCORES FOR THE CURRENT DRIVE, USING DATA WRITTEN BY ALL OTHER DRIVES. THE SCORES ARE CONVERTED ON THE BASIS OF 12(DEC), (WITH A PERFECT SCORE = 12). ALSO, THE SCORES ARE LISTED SEPARATELY FOR EACH TRACK (SURFACE), FOR EACH OFFSET DIRECTION, AND AN OVERALL AVERAGE FOR ALL THREE TRACKS, ON THE CURRENT DRIVE, IS PRINTED.

IN THE FOLLOWING EXAMPLE, THERE ARE 2 SYSTEMS, AND THE DRIVES BEING TESTED ARE A0,A1,A2,B0, AND B5. THE TEST RESULTS FOR DRIVE A1 ARE SHOWN BELOW:

SCORES FOR DRIVE A1 (0-12 DEC.) :

TRACK NO.	DRIVE READ	OVRWRT SCORE OFST-	OVRWRT SCORE OFST+	READ SCORE OFST-	READ SCORE OFST+
0	A0	12	12	12	12
0	SELF	12	12	12	12
0	A2	* 7	* 6	12	12
0	B0	11	12	11	11
0	B5	9	12	11	11
1	A0	12	11	12	11
1	SELF	11	10	12	12
1	A2	12	11	11	12
1	B0	12	12	9	9
1	B5	12	12	11	12
2	A0	12	12	12	12
2	SELF	12	12	12	12
2	A2	11	9	11	11
2	B0	12	12	12	12
2	B5	12	12	12	12

DRIVE A1 OVWRT AVG = 12
DRIVE A1 READ AVG = 10

THE ABOVE EXAMPLE REVEALS A POSSIBLE COMPATIBILITY PROBLEM BETWEEN DRIVES A1 AND A2. NOTICE THAT ON TRACK 0, THAT THE OVERWRITE SCORES

WERE UNACCEPTABLY LOW (7 OR LESS), AND THE PROGRAM NOTED THESE BAD SCORES WITH AN ASTERISK (*).

11.0 ERROR REPORTING

11.1 COMMON ERRORS

THE FOLLOWING IS A LIST OF COMMON ERROR MESSAGES WHICH ACCOMPANY ERROR TYPEOUTS FROM THE RK06 DRIVE COMPATIBILITY PROGRAM. THE ERRORS ARE SELF-EXPLANATORY.

1. UNIBUS PARITY ERROR
2. NON-EXISTANT MEMORY ERROR
3. NON-EXISTANT DRIVE ERROR
4. UNIT FIELD ERROR
5. SUBSYSTEM TIMEOUT
6. SERCON PARITY ERROR
7. DRIVE DETECTED PARITY ERROR
8. AC LOW
9. SPEED LOSS
10. ILLEGAL FUNCTION ERROR
11. PROGRAMMING ERROR
12. NON-EXECUTABLE FUNCTION ERROR
13. DRIVE TYPE ERROR
14. FORMAT ERROR
15. WRITE LOCK ERROR
16. DRIVE UNSAFE ERROR
17. SEEK INCOMPLETE ERROR
18. CYLINDER OVERFLOW ERROR
19. ILLEGAL CYLINDER ADDRESS ERROR
20. DRIVE OFF TRACK
21. DRIVE TIMING ERROR

```
919
920
921
922                                     22.  DATA LATE ERROR
923
924                                     23.  CONTROLLER TIMEOUT ERROR
925
926                                     24.  OPERATION INCOMPLETE ERROR
927
928                                     25.  HEADER VRC ERROR
929
930                                     26.  DATA CHECK ERROR
931
932                                     27.  WRITE CHECK ERROR
933
934                                     28.  DATA MISCOMPARE
935
936                                     29.  NO DRIVE RESPONSE - UFE AND NXD
937
938                                     30.  DRIVE ERROR WILL NOT CLEAR
939
940                                     31.  DRIVE STATUS CHANGE WILL NOT CLEAR
941
942                                     32.  ATTENTION BUT NO STATUS CHANGE OR FAULT
943
944                                     33.  ATTENTION BUT DRIVE NOT AVAILABLE
945
946                                     34.  ERROR WHILE GATHERING DRIVE STATUS
947
948                                     35.  MULTIPLE DRIVE SELECT
949
950                                     36.  HEADER COMPARE ERROR
951
952                                     37.  ERROR IN RECALIBRATE FOR RECOVERY
953
954                                     38.  CLEAR CONTROLLER DID NOT CLEAR ERROR
955
956                                     39.  NO ATTENTION IN ATTENTION SUMMARY REGISTER
957
958                                     40.  UNSOLICITED ATTENTION
959
960                                     41.  UNEXPECTED DATA TYPE ERROR
961
962                                     42.  ATTENTION DID NOT RESET WITH CLEAR
963
964                                     43.  SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
965
966                                     44.  DATA LATE WHEN UNLOADING HEADER
967
968                                     45.  CONTROLLER ERROR WHEN DRIVER SERVICING
969
970                                     46.  RETRY UNSUCCESSFUL
971
972                                     47.  BAD SECTOR ERROR ON SECTOR NOT LISTED BAD
973
```

11.2 ERROR HANDLING

ERRORS REPORTED BY THE PROGRAM CONSIST OF COMMON FAILURES RESULTING FROM ATTEMPTED SUBSYSTEM FUNCTIONS, AS WELL AS CERTAIN ERRORS UNIQUE TO PARTICULAR TESTS. EACH ERROR PRINTOUT CONSISTS OF AN ERROR DESCRIPTION AND TEST NUMBER, POSSIBLY FOLLOWED BY HEADER LINES, COLUMN HEADINGS, AND COLUMNS OF REGISTER CONTENTS IN OCTAL. AS MUCH MEANINGFUL REGISTER DATA AS POSSIBLE (FOR EXAMPLE, RK611 REGISTERS) ARE REPORTED IN A GIVEN ERROR. OTHER ERROR REPORTS MAY CONSIST OF A SINGLE DESCRIPTIVE LINE.

11.3 ERROR PRINTOUT EXAMPLE

```
     ** WRITE CHECK ERROR

     PREVIOUS COMMAND:
     DRIVE     CMND     CYLNDR     TRACK     SECTOR     WD CNT
     000000    000121   000016     000001    000000     175000
       HI BA     LO BA
       000000    061566

     CURRENT COMMAND:
     ERR PC      DRIVE     CMND     CYLNDR     TRACK     SECTOR     WD CNT
     041154      000000    000131   000016     000001    000006     175000
       HI BA     LO BA
       000000    061566

     PACK ADDRESS OF ERROR(S):
     CYLNDR     TRACK     SECTOR
     000016     000001    000006
```

TABLE A

BASIC READ/WRITE TEST SECTORS

ADDRESS OF SECTOR ON EACH SURFACE (IN OCTAL)

DRIVE NO.	CYLINDER	SECTOR
0	620	0
1	620	2
2	620	4
3	620	6
4	620	10
5	620	12
6	620	14
7	620	16
10	622	0
11	622	2
12	622	4
13	622	6
14	622	10
15	622	12
16	622	14
17	622	16

TABLE B

WORST CASE DATA PATTERN (REPEATS EVERY 16 WORDS)

WORD NO.	DATA (OCTAL)
0	072307
1	135143
2	156461
3	167230
4	073514
5	035646
6	016723
7	107351
10	143564
11	061672
12	030735
13	114356
14	046167
15	123073
16	151453
17	164616

TABLE C

CYLINDER BLOCK ASSIGNMENT FOR A GIVEN SURFACE

CURRENT ZONE - RANGE	OVERWRITE CYL BLK RANGE (OCT)	COMPATIBILITY CYL BLK RANGE (OCT)	ROTATING STARTING SECTOR
1 - CYL 0-77	CYL 0-17	CYL 60-77	SECT 0
2 - 100-177	100-117	160-177	3
3 - 200-277	200-217	260-277	7
4 - 300-377	300-317	360-377	13
5 - 400-477	400-417	460-477	17
6 - 500-577	500-517	560-577	22
7 - 600-632	600-617	---	0

TABLE D

BASIC CYLINDER BLOCK LAYOUT EXAMPLE

CYLINDER NUMBERS (OCTAL)	SECTOR NUMBERS (OCTAL)															
	0	1	2	3	5	6	7	10	12	13	14	15	17	20	21	22
60	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
61	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	0
62	3	4	5	6	7	10	11	12	13	14	15	16	17	0	1	2
63	6	7	10	11	12	13	14	15	16	17	0	1	2	3	4	5
64	12	13	14	15	16	17	0	1	2	3	4	5	6	7	10	11
65	17	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16
66	5	6	7	10	11	12	13	14	15	16	17	0	1	2	3	4
67	14	15	16	17	0	1	2	3	4	5	6	7	10	11	12	13
70	4	5	6	7	10	11	12	13	14	15	16	17	0	1	2	3
71	15	16	17	0	1	2	3	4	5	6	7	10	11	12	13	14
72	7	10	11	12	13	14	15	16	17	0	1	2	3	4	5	6
73	2	3	4	5	6	7	10	11	12	13	14	15	16	17	0	1
74	16	17	0	1	2	3	4	5	6	7	10	11	12	13	14	15
75	13	14	15	16	17	0	1	2	3	4	5	6	7	10	11	12
76	11	12	13	14	15	16	17	0	1	2	3	4	5	6	7	10
77	10	11	12	13	14	15	16	17	0	1	2	3	4	5	6	7

TABLE E

OVERWRITE CYLINDERS

DRIVE #	CYLINDERS OVERWRITTEN (OCTAL)
0	0,100,200,300,400,500,600
1	1,101,201,301,401,501,601
2	2,102,202,302,402,502,602
3	3,103,203,303,403,503,603
4	4,104,204,304,404,504,604
5	5,105,205,305,405,505,605
6	6,106,206,306,406,506,606
7	7,107,207,307,407,507,607
10	10,110,210,310,410,510,610
11	11,111,211,311,411,511,611
12	12,112,212,312,412,512,612
13	13,113,213,313,413,513,613
14	14,114,214,314,414,514,614
15	15,115,215,315,415,515,615
16	16,116,216,316,416,516,616
17	17,117,217,317,417,517,617

TABLE F

SELF-TEST CYLINDERS

DRIVE #	CYLINDERS (WHERE SECTORS 4,11,16,23 ARE TESTED)
0	60,160,260,360,460,560
1	61,161,261,361,461,561
2	62,162,262,362,462,562
3	63,163,263,363,463,563
4	64,164,264,364,464,564
5	65,165,265,365,465,565
6	66,166,266,366,466,566
7	67,167,267,367,467,567
10	70,170,270,370,470,570
11	71,171,271,371,471,571
12	72,172,272,372,472,572
13	73,173,273,373,473,573
14	74,174,274,374,474,574
15	75,175,275,375,475,575
16	76,176,276,376,476,576
17	77,177,277,377,477,577

TABLE G

PSEUDO-RANDOM DATA PATTERN

WORD #	DATA (OCTAL)
0	040135
1	177070
2	070414
3	064531
4	174473
5	062422
6	114352
7	036620
10	010031
11	052336
12	017310
13	011347
14	102367
15	152567
16	001246
17	160073

```
                                .NLIST MC,MD,CND
                                .LIST  ME
                                .ENABL ABS,AMA



                        ;;*****************************************************************
                        .SBTTL  STARTING ADDRESSES
                        ;*
                        ;*      200     DEFAULT PARAMETERS AND RUN PASS 1 AND PASS 2
                        ;*      204     SELECT PARAMETERS AND RUN PASS 1
                        ;*      220     SELECT PARAMETERS AND RUN PASS 2
                        ;*      224     MEMORY DUMP ROUTINE
                        ;;*****************************************************************


000001                  $TN=1
                        .TITLE  MD-11-DZR6Q-A - RK06 DRIVE COMPATIBILITY PROGRAM
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DAVE HOFFMAN
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
160000                  $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT

163000                          $SWR=163000
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            REPORT DESCRIPTION ONLY, ON ERRORS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            APPLY RANDOM STALL BETWEEN OPERATIONS
                        ;*         7            TYPE BAD SECTOR FILES (BSF'S) AT START

                        .SBTTL  BASIC DEFINITIONS

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

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
```

```
1311            000200          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
1312            177776          PS=     177776          ;;PROCESSOR STATUS WORD
1313                            .EQUIV  PS,PSW
1314            177774          STKLMT= 177774          ;;STACK LIMIT REGISTER
1315            177772          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
1316            177570          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
1317            177570          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
1318
1319                            ;*GENERAL PURPOSE REGISTER DEFINITIONS
1320            000000          R0=     %0              ;;GENERAL REGISTER
1321            000001          R1=     %1              ;;GENERAL REGISTER
1322            000002          R2=     %2              ;;GENERAL REGISTER
1323            000003          R3=     %3              ;;GENERAL REGISTER
1324            000004          R4=     %4              ;;GENERAL REGISTER
1325            000005          R5=     %5              ;;GENERAL REGISTER
1326            000006          R6=     %6              ;;GENERAL REGISTER
1327            000007          R7=     %7              ;;GENERAL REGISTER
1328            000006          SP=     %6              ;;STACK POINTER
1329            000007          PC=     %7              ;;PROGRAM COUNTER
1330
1331                            ;*PRIORITY LEVEL DEFINITIONS
1332            000000          PR0=    0               ;;PRIORITY LEVEL 0
1333            000040          PR1=    40              ;;PRIORITY LEVEL 1
1334            000100          PR2=    100             ;;PRIORITY LEVEL 2
1335            000140          PR3=    140             ;;PRIORITY LEVEL 3
1336            000200          PR4=    200             ;;PRIORITY LEVEL 4
1337            000240          PR5=    240             ;;PRIORITY LEVEL 5
1338            000300          PR6=    300             ;;PRIORITY LEVEL 6
1339            000340          PR7=    340             ;;PRIORITY LEVEL 7
1340
1341                            ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1342            100000          SW15=   100000
1343            040000          SW14=   40000
1344            020000          SW13=   20000
1345            010000          SW12=   10000
1346            004000          SW11=   4000
1347            002000          SW10=   2000
1348            001000          SW09=   1000
1349            000400          SW08=   400
1350            000200          SW07=   200
1351            000100          SW06=   100
1352            000040          SW05=   40
1353            000020          SW04=   20
1354            000010          SW03=   10
1355            000004          SW02=   4
1356            000002          SW01=   2
1357            000001          SW00=   1
1358                            .EQUIV  SW09,SW9
1359                            .EQUIV  SW08,SW8
1360                            .EQUIV  SW07,SW7
1361                            .EQUIV  SW06,SW6
1362                            .EQUIV  SW05,SW5
1363                            .EQUIV  SW04,SW4
1364                            .EQUIV  SW03,SW3
1365                            .EQUIV  SW02,SW2
1366                            .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
                        000030  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034  TRAPVEC=34              ;;"TRAP" TRAP
                        000060  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064  TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                                .SBTTL  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  STARTING ADDRESS(ES)
000200  000137  011746          JMP     @#DFSTRT        ;;JUMP TO STARTING ADDRESS OF PROGRAM
                        000204          .=204
```

```
1423  000204  000137  011754          JMP     @#P1STRT
1424          000220                  .=220
1425  000220  000137  011772          JMP     @#P2STRT
1426
1427  000224  000700          ..LOW:  .WORD   700
1428  000226  001100          ..HIGH: .WORD   1100
1429
1430                          .SBTTL  ACT11 HOOKS
1431
1432                          ;;******************************************************************
1433                          ;HOOKS REQUIRED BY ACT11
1434          000230                  $SVPC=.                 ;SAVE PC
1435          000046                  .=46
1436  000046  016712                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1437          000052                  .=52
1438  000052  140000                  .WORD   140000          ;;2)SET LOC.52 TO 140000
1439          000230                  .=$SVPC                 ;; RESTORE PC
1440          001000                  .=1000
1441                          .SBTTL  APT PARAMETER BLOCK
1442
1443                          ;;******************************************************************
1444                          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1445                          ;;******************************************************************
1446          001000                  .$X=.   ;;SAVE CURRENT LOCATION
1447          000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1448  000024  000200                  200     ;;FOR APT START UP
1449          000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1450  000044  001000                  $APTHDR ;;POINT TO APT HEADER BLOCK
1451          001000                  .=.$X   ;;RESET LOCATION COUNTER
1452                          ;;******************************************************************
1453                          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1454                          ;INTERFACE SPEC.
1455
1456  001000                  $APTHD:
1457  001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1458  001002  001330          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1459  001004  001320          $TSTM:  .WORD   1320    ;;RUN TIM OF LONGEST TEST
1460  001006  001546          $PASTM: .WORD   1546    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1461  001010  001546          $UNITM: .WORD   1546    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1462  001012  000030                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1463          120210          AVECT1=120210           ;RKVEC=210, RKPRI=5
1464          177440          ABASE=177440            ;RKBAS ADRS
1465          000377          ADEVM=000377            ;SET DEVICES 0-7 IN MAP
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
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                  $REG11: .WORD   0       ;;CONTAINS (($REGAD)+22)
001206  000000                  $REG12: .WORD   0       ;;CONTAINS (($REGAD)+24)
001210  000000                  $REG13: .WORD   0       ;;CONTAINS (($REGAD)+26)
001212  000000                  $REG14: .WORD   0       ;;CONTAINS (($REGAD)+30)
001214  000000                  $REG15: .WORD   0       ;;CONTAINS (($REGAD)+32)
001216  000000                  $REG16: .WORD   0       ;;CONTAINS (($REGAD)+34)
001220  000000                  $REG17: .WORD   0       ;;CONTAINS (($REGAD)+36)
001222  000000                  $REG20: .WORD   0       ;;CONTAINS (($REGAD)+40)
```

```
1522  001224  000000                $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
1523  001226  000000                $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
1524  001230  000000                $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
1525  001232  000000                $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
1526  001234  000000                $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
1527  001236  000000                $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
1528  001240  000000                $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
1529  001242  000000                $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
1530  001244  000000                $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
1531  001246  000000                $REG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
1532  001250  000000                $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
1533  001252  000000                $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
1534  001254  000000                $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
1535  001256  000000                $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
1536  001260  000000                $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
1537  001262  000000                $TMP0:  .WORD   0               ;;USER DEFINED
1538  001264  000000                $TMP1:  .WORD   0               ;;USER DEFINED
1539  001266  000000                $TMP2:  .WORD   0               ;;USER DEFINED
1540  001270  000000                $TMP3:  .WORD   0               ;;USER DEFINED
1541  001272  000000                $TMP4:  .WORD   0               ;;USER DEFINED
1542  001274  000000                $TMP5:  .WORD   0               ;;USER DEFINED
1543  001276  000000                $TMP6:  .WORD   0               ;;USER DEFINED
1544  001300  000000                $TMP7:  .WORD   0               ;;USER DEFINED
1545  001302  000000                $TMP10: .WORD   0               ;;USER DEFINED
1546  001304  000000                $TMP11: .WORD   0               ;;USER DEFINED
1547  001306  000000                $TMP12: .WORD   0               ;;USER DEFINED
1548  001310  000000                $TMP13: .WORD   0               ;;USER DEFINED
1549  001312  000000                $TMP14: .WORD   0               ;;USER DEFINED
1550  001314  000000                $TMP15: .WORD   0               ;;USER DEFINED
1551  001316  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
1552  001320  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
1553  001324     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
1554  001325     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1555  001326  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
1556                                ;;**************************************************************
1557                                .SBTTL  APT MAILBOX-ETABLE
1558
1559                                ;;**************************************************************
1560                                .EVEN
1561  001330                        $MAIL:                          ;;APT MAILBOX
1562  001330  000000                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
1563  001332  000000                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
1564  001334  000000                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
1565  001336  000000                $PASS:  .WORD   APASS           ;;PASS COUNT
1566  001340  000000                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
1567  001342  000000                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
1568  001344  000000                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
1569  001346  000000                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
1570  001350                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
1571  001350     000                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1572  001351     000                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1573  001352  000000                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1574  001354  000000                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1575  001356  000000                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1576                                ;*                              BITS 15-11=CPU TYPE
1577                                ;*                                   11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
```

```
1578                                    ;*                      11/70=06,PDQ=07,Q=10
1579                                    ;*                  BIT 10=REAL TIME CLOCK
1580                                    ;*                  BIT  9=FLOATING POINT PROCESSOR
1581                                    ;*                  BIT  8=MEMORY MANAGEMENT
1582  001360      000                   $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1583  001361      000                   $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1584                                    ;*                  MEM.TYPE BYTE   --  (HIGH BYTE)
1585                                    ;*                          900 NSEC CORE=001
1586                                    ;*                          300 NSEC BIPOLAR=002
1587                                    ;*                          500 NSEC MOS=003
1588  001362  000000                    $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1589                                    ;*                  MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1590  001364      000                   $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1591  001365      000                   $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1592  001366  000000                    $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1593  001370      000                   $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1594  001371      000                   $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1595  001372  000000                    $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1596  001374      000                   $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1597  001375      000                   $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1598  001376  000000                    $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1599  001400  120210                    $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1600  001402  000000                    $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1601  001404  177440                    $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1602  001406  000377                    $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1603  001410                            $ETEND:
1604                                    .MEXIT
```

```
                                .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

001410                          $ERRTB:

                                        ;ERROR 1                ;UNIBUS PARITY ERROR
001410  044553                         EM1
001412  047646                         DH100
001414  051432                         DT100
001416  051542                         DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
001420  044577                         EM2
001422  047646                         DH100
001424  051432                         DT100
001426  051542                         DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
001430  044623                         EM3
001432  047646                         DH100
001434  051432                         DT100
001436  051542                         DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
001440  044646                         EM4
001442  047646                         DH100
001444  051432                         DT100
001446  051542                         DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
001450  044667                         EM5
001452  047646                         DH100
001454  051432                         DT100
001456  051572                         DF02

                                        ;ERROR 6
001460  044706                         EM6                     ;D TO C PARITY ERROR
001462  047646                         DH100
001464  051432                         DT100
001466  051626                         DF03

                                        ;ERROR 7
001470  044732                         EM7                     ;DRIVE DETECTED PARITY ERROR
001472  047646                         DH100
001474  051432                         DT100
```

```
1661  001476  051626                    DF03
1662
1663                                    ;ERROR 10
1664  001500  044766                    EM10                ;AC LOW
1665  001502  047646                    DH100
1666  001504  051432                    DT100
1667  001506  051572                    DF02
1668
1669                                    ;ERROR 11
1670  001510  044775                    EM11                ;SPEED LOSS
1671  001512  047646                    DH100
1672  001514  051432                    DT100
1673  001516  051572                    DF02
1674
1675                                    ;ERROR 12
1676  001520  045010                    EM12                ;ILLEGAL FUNCTION
1677  001522  047646                    DH100
1678  001524  051432                    DT100
1679  001526  051572                    DF02
1680
1681                                    ;ERROR 13
1682  001530  045031                    EM13                ;PROGRAMMING ERROR
1683  001532  047646                    DH100
1684  001534  051432                    DT100
1685  001536  051542                    DF01
1686
1687                                    ;ERROR 14           ;NON-EXISTANT FUNCTION
1688  001540  045053                    EM14
1689  001542  047646                    DH100
1690  001544  051432                    DT100
1691  001546  051572                    DF02
1692
1693                                    ;ERROR 15
1694  001550  045101                    EM15                ;DRIVE TYPE ERROR
1695  001552  047646                    DH100
1696  001554  051432                    DT100
1697  001556  051572                    DF02
1698
1699                                    ;ERROR 16
1700  001560  045122                    EM16                ;FORMAT ERROR
1701  001562  047646                    DH100
1702  001564  051432                    DT100
1703  001566  051572                    DF02
1704
1705                                    ;ERROR 17
1706  001570  045137                    EM17                ;WRITE LOCK ERROR
1707  001572  047646                    DH100
1708  001574  051432                    DT100
1709  001576  051572                    DF02
1710
1711                                    ;ERROR 20
1712  001600  045160                    EM20                ;DRIVE UNSAFE
1713  001602  047646                    DH100
1714  001604  051432                    DT100
1715  001606  051572                    DF02
1716
```

```
1717                                    ;ERROR 21
1718  001610  045175                    EM21                ;SEEK INCOMPLETE
1719  001612  047646                    DH100
1720  001614  051432                    DT100
1721  001616  051572                    DF02
1722
1723                                    ;ERROR 22
1724  001620  045215                    EM22                ;CYLINDER OVERFLOW
1725  001622  047646                    DH100
1726  001624  051432                    DT100
1727  001626  051572                    DF02
1728
1729                                    ;ERROR 23
1730  001630  045237                    EM23                ;ILLEGAL CYLINDER
1731  001632  047646                    DH100
1732  001634  051432                    DT100
1733  001636  051572                    DF02
1734
1735                                    ;ERROR 24
1736  001640  045270                    EM24                ;DRIVE OFF TRACK
1737  001642  047646                    DH100
1738  001644  051432                    DT100
1739  001646  051572                    DF02
1740
1741                                    ;ERROR 25
1742  001650  045310                    EM25                ;DRIVE TIMING ERROR
1743  001652  047646                    DH100
1744  001654  051432                    DT100
1745  001656  051572                    DF02
1746
1747                                    ;ERROR 26
1748  001660  045333                    EM26                ;DATA LATE
1749  001662  047646                    DH100
1750  001664  051432                    DT100
1751  001666  051572                    DF02
1752
1753                                    ;ERROR 27
1754  001670  045345                    EM27                ;CONTROLLER TIMEOUT
1755  001672  047646                    DH100
1756  001674  051432                    DT100
1757  001676  051572                    DF02
1758
1759                                    ;ERROR 30           ;OPERATION INCOMPLETE
1760  001700  045370                    EM30
1761  001702  047646                    DH100
1762  001704  051432                    DT100
1763  001706  051702                    DF05
1764
1765                                    ;ERROR 31
1766  001710  045415                    EM31                ;HEADER VRC ERROR
1767  001712  047646                    DH100
1768  001714  051432                    DT100
1769  001716  051702                    DF05
1770
1771                                    ;ERROR 32
1772  001720  045436                    EM32                ;DATA CHECK ERROR
```

```
1773  001722  047646                  DH100
1774  001724  051432                  DT100
1775  001726  051736                  DF07
1776
1777                                  ;ERROR 33
1778  001730  045457                  EM33                ;WRITE CHECK ERROR
1779  001732  047646                  DH100
1780  001734  051432                  DT100
1781  001736  051772                  DF10
1782
1783                                  ;ERROR 34
1784  001740  045501                  EM34                ;DATA MISCOMPARE(S)
1785  001742  047646                  DH100
1786  001744  051432                  DT100
1787  001746  051666                  DF04
1788
1789                                  ;ERROR 35
1790  001750  045521                  EM35                ;NO DRIVE RESPONSE-UFE & NXD
1791  001752  047646                  DH100
1792  001754  051432                  DT100
1793  001756  051542                  DF01
1794
1795                                  ;ERROR 36
1796  001760  045555                  EM36                ;DRIVE ERROR WILL NOT CLEAR
1797  001762  000000                  0
1798  001764  000000                  0
1799  001766  000000                  0
1800
1801                                  ;ERROR 37           ;DRIVE STATUS CHANGE WILL NOT CLEAR
1802  001770  045610                  EM37
1803  001772  000000                  0
1804  001774  000000                  0
1805  001776  000000                  0
1806
1807                                  ;ERROR 40
1808  002000  045653                  EM40                ;ATTENTION BUT NO STATUS CHANGE OR FAULT
1809  002002  047646                  DH100
1810  002004  051432                  DT100
1811  002006  051572                  DF02
1812
1813                                  ;ERROR 41
1814  002010  045717                  EM41                ;ATTENTION BUT DRIVE NOT AVAILABLE
1815  002012  047646                  DH100
1816  002014  051432                  DT100
1817  002016  051572                  DF02
1818
1819                                  ;ERROR 42
1820  002020  045755                  EM42                ;ATTENTION WHEN NOT EXPECTED
1821  002022  047646                  DH100
1822  002024  051432                  DT100
1823  002026  051572                  DF02
1824
1825                                  ;ERROR 43           ;ERROR WHILE GATHERING DRIVE STATUS
1826  002030  046005                  EM43
1827  002032  047646                  DH100
1828  002034  051432                  DT100
```

```
1829  002036  052036                  DF12
1830
1831                                  ;ERROR 44
1832  002040  046262                  EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
1833  002042  047646                  DH100
1834  002044  051432                  DT100
1835  002046  052036                  DF12
1836
1837                                  ;ERROR 45
1838  002050  046327                  EM64                ;NO ATTENTION IN ATTENTION SUMMARY REG
1839  002052  047646                  DH100
1840  002054  051432                  DT100
1841  002056  052036                  DF12
1842
1843                                  ;ERROR 46
1844  002060  046372                  EM65                ;UNSOLICITED ATTENTION
1845  002062  047646                  DH100
1846  002064  051432                  DT100
1847  002066  052036                  DF12
1848
1849                                  ;ERROR 47
1850  002070  046420                  EM66                ;UNEXPECTED DATA TYPE ERROR
1851  002072  047646                  DH100
1852  002074  051432                  DT100
1853  002076  052036                  DF12
1854
1855                                  ;ERROR 50
1856  002100  046453                  EM67                ;ATTENTION DID NOT RESET WITH CLEAR
1857  002102  047646                  DH100
1858  002104  051432                  DT100
1859  002106  052036                  DF12
1860
1861                                  ;ERROR 51
1862  002110  046512                  EM70                ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1863  002112  047646                  DH100
1864  002114  051432                  DT100
1865  002116  052036                  DF12
1866
1867                                  ;ERROR 52
1868  002120  046042                  EM52                ;MULTIPLE DRIVE SELECT
1869  002122  047646                  DH100
1870  002124  051432                  DT100
1871  002126  052036                  DF12
1872
1873                                  ;ERROR 53
1874  002130  046070                  EM53                ;ABREVIATED HCE ERROR
1875  002132  047646                  DH100
1876  002134  051432                  DT100
1877  002136  052066                  DF13
1878
1879                                  ;ERROR 54
1880  002140  045370                  EM30                ;OPERATION INCOMPLETE ERROR
1881  002142  047646                  DH100
1882  002144  051432                  DT100
1883  002146  052116                  DF14
1884
```

```
1885
1886  002150  045415          ;ERROR 55
                              EM31                ;ABREVIATED HVRC ERROR
1887  002152  047646          DH100
1888  002154  051432          DT100
1889  002156  052066          DF13
1890
1891                          ;ERROR 56
1892  002160  046115          EM56                ;2 TIMEOUT ERROR
1893  002162  047646          DH100
1894  002164  051432          DT100
1895  002166  052146          DF15
1896
1897                          ;ERROR 57           ;2ND LEVEL IN SUBSYSTEM TIMEOUT
1898  002170  046115          EM56
1899  002172  047646          DH100
1900  002174  051432          DT100
1901  002176  052212          DF16
1902
1903                          ;ERROR 60
1904  002200  046134          EM60                ;ERROR IN RECAL FOR RECOVERY
1905  002202  000000          0
1906  002204  000000          0
1907  002206  000000          0
1908
1909                          ;ERROR 61
1910  002210  046170          EM61                ;ABORT MESSAGE
1911  002212  000000          0
1912  002214  000000          0
1913  002216  000000          0
1914
1915                          ;ERROR 62
1916  002220  046236          EM62                ;CYLINDER MISCOMPARE
1917  002222  047646          DH100
1918  002224  051432          DT100
1919  002226  052256          DF17
1920
1921                          ;ERROR 63           ;DATA ERROR WORDS
1922  002230  000000          0
1923  002232  000000          0
1924  002234  051524          DT602
1925  002236  052366          DF25
1926
1927                          ;ERROR 64
1928  002240  046262          EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
1929  002242  047646          DH100
1930  002244  051432          DT100
1931  002246  051572          DF02
1932
1933                          ;ERROR 65
1934  002250  046327          EM64                ;NO ATTENTION IN ATTENTION SUMMARY REG
1935  002252  047646          DH100
1936  002254  051432          DT100
1937  002256  051572          DF02
1938
1939                          ;ERROR 66
1940  002260  046372          EM65                ;UNSOLICITED ATTENTION
```

```
1941  002262  047646                  DH100
1942  002264  051432                  DT100
1943  002266  051572                  DF02
1944
1945                                  ;ERROR 67
1946  002270  046420                  EM66          ;UNEXPECTED DATA TYPE ERROR
1947  002272  047646                  DH100
1948  002274  051432                  DT100
1949  002276  051572                  DF02
1950
1951                                  ;ERROR 70
1952  002300  046453                  EM67          ;ATTENTION DID NOT RESET WITH CLEAR
1953  002302  047646                  DH100
1954  002304  051432                  DT100
1955  002306  051572                  DF02
1956
1957                                  ;ERROR 71
1958  002310  046512                  EM70          ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
1959  002312  047646                  DH100
1960  002314  051432                  DT100
1961  002316  051572                  DF02
1962
1963                                  ;ERROR 72
1964  002320  046561                  EM71          ;DATA LATE WHEN UNLOADING HEADER
1965  002322  047646                  DH100
1966  002324  051432                  DT100
1967  002326  051572                  DF02
1968
1969                                  ;ERROR 73
1970  002330  046621                  EM72          ;CONTROLLER ERROR DURING DRIVER SERVICE
1971  002332  047646                  DH100
1972  002334  051432                  DT100
1973  002336  051572                  DF02
1974
1975                                  ;ERROR 74
1976  002340  046670                  EM73          ;DRIVE DETECTED PARITY ERROR
1977  002342  047646                  DH100
1978  002344  051432                  DT100
1979  002346  051572                  DF02
1980
1981                                  ;ERROR 75
1982  002350  046724                  EM74          ;UNDEFINED ERROR
1983  002352  047646                  DH100
1984  002354  051432                  DT100
1985  002356  051572                  DF02
1986
1987                                  ;ERROR 76
1988  002360  046744                  EM75          ;MARKING SECTOR BAD MESSAGE
1989  002362  000000                  0
1990  002364  000000                  0
1991  002366  000000                  0
1992
1993                                  ;ERROR 77
1994  002370  046774                  EM76          ;BAD DATA VERIFICATION WITH READ
1995  002372  050633                  DH605
1996  002374  051516                  DT601
```

```
1997  002376  052312                        DF21
1998
1999                                        ;ERROR 100
2000  002400  047056                        EM77                ;RETRY SUCCESFUL MESSAGE
2001  002402  000000                        0
2002  002404  051516                        DT601
2003  002406  052352                        DF23
2004
2005                                        ;ERROR 101
2006  002410  047056                        EM77                ;ANOTHER RETRY SUCCESSFUL MESSAGE
2007  002412  000000                        0
2008  002414  051516                        DT601
2009  002416  052352                        DF23
2010
2011                                        ;ERROR 102
2012  002420  047077                        EM100               ;RETRY UNSUCCESSFUL MESSAGE
2013  002422  000000                        0
2014  002424  051516                        DT601
2015  002426  052352                        DF23
2016
2017                                        ;ERROR 103
2018  002430  047122                        EM101               ;NO VALID HEADERS IN TRACK JUST READ
2019  002432  050615                        DH6042
2020  002434  051516                        DT601
2021  002436  052362                        DF24
2022
2023                                        ;ERROR 104
2024  002440  047200                        EM102               ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2025  002442  047646                        DH100
2026  002444  051432                        DT100
2027  002446  051572                        DF02
2028
2029                                        ;ERROR  105
2030  002450  047252                        EM103               ;TIMED-OUT ON READ HEADER
2031  002452  047646                        DH100
2032  002454  051432                        DT100
2033  002456  051626                        DF03
2034
2035                                        ;ERROR  106
2036  002460  047300                        EM104               ;TIMED-OUT ON SEEK
2037  002462  047646                        DH100
2038  002464  051432                        DT100
2039  002466  051626                        DF03
2040
2041                                        ;ERROR  107
2042  002470  047322                        EM105               ;DRIVE SIEZED BY OTHER PORT
2043  002472  047646                        DH100
2044  002474  051432                        DT100
2045  002476  051572                        DF02
2046
2047                                        ;ERROR  110
2048  002500  047355                        EM106               ;"DATA MISCMPR WHILE BAI SET"
2049  002502  047646                        DH100
2050  002504  051432                        DT100
2051  002506  052402                        DF27
2052
```

```
2053                                    ;ERROR  111
2054  002510  047410                    EM107                   ;"NO NEM WHEN EXPECTED"
2055  002512  000000                    0
2056  002514  000000                    0
2057  002516  000000                    0
2058
2059                                    ;ERROR  112
2060  002520  047455                    EM110                   ;"INTRPT WHEN CNTRLR NOT READY"
2061  002522  047646                    DH100
2062  002524  051432                    DT100
2063  002526  051542                    DF01
2064
2065                                    ;ERROR  113
2066  002530  047510                    EM111                   ;"NO ATT'N ON SEEK"
2067  002532  047646                    DH100
2068  002534  051432                    DT100
2069  002536  051572                    DF02
2070
2071                                    ;ERROR  114
2072  002540  047531                    EM112                   ;DRIVE'S CYLINDER INCORRECT
2073  002542  051042                    DH702
2074  002544  051472                    DT202
2075  002546  052372                    DF26
2076
2077                                    ;ERROR  115
2078  002550  000000                    0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2079  002552  000000                    0
2080  002554  051516                    DT601
2081  002556  052016                    DF11
2082
2083                                    ;ERROR  116
2084  002560  045501                    EM34                    ;DATA MISCOMPARE (11/70)
2085  002562  050157                    DH103
2086  002564  000000                    0
2087  002566  052302                    DF20
2088
2089                                    ;ERROR  117
2090  002570  000000                    0                       ;PART OF DATA MISCOMPARE
2091  002572  000000                    0
2092  002574  051432                    DT100
2093  002576  052322                    DF22
2094
2095                                    ;ERROR  120
2096  002600  047564                    EM113                   ;ABORT- CAN'T READ BSF
2097  002602  047646                    DH100
2098  002604  051432                    DT100
2099  002606  051542                    DF01
2100
2101                                    ;ERROR  121
2102  002610  047612                    EM114                   ;KT11 FAILURE
2103  002612  047646                    DH100
2104  002614  051432                    DT100
2105  002616  052426                    DF30
2106
2107                                    ;ERROR  122
2108  002620  047627                    EM115                   ;MEM PARITY ERROR
```

```
2109  002622  047646  DH100
2110  002624  051432  DT100
2111  002626  052452  DF31
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
```

```
                        .SBTTL  BIT ASSIGNMENTS IN THE RK611 REGISTERS

                        ;RKCS1
                        ; 15      14      13      12      11      10      9       8
                        ;-----------------------------------------------------------------
                        ;CERR !  DI   ! DCPAR! CFMT  ! CTO   ! CDT   ! BA17  ! BA16  !
                        ;CCLR !       !       !       !       !       !       !       !
                        ; R/W  !  RO   !  RO   !  R/W  !  RO   !  R/W  !  R/W  !  R/W  !
                        ;-----------------------------------------------------------------

                        ;  7      6       5       4       3       2       1       0
                        ;-----------------------------------------------------------------
                        ; RDY  !  IE   ! SPARE!  F4   !  F3   !  F2   !  F1   !  GO   !
                        ; R/W  !  R/W  !  R/W  !  R/W  !  R/W  !  R/W  !  R/W  !  R/W  !
                        ;-----------------------------------------------------------------

                        ;RKCS2
                        ; 15      14      13      12      11      10      9       8
                        ;-----------------------------------------------------------------
                        ; DLT  !  WCE  !  UPE  !  NED  !  NEM  !  PGE  !  MDS  !  UFE  !
                        ; RO   !  RO   !  RO   !  RO   !  RO   !  RO   !  RO   !  RO   !
                        ;-----------------------------------------------------------------

                        ;  7      6       5       4       3       2       1       0
                        ;-----------------------------------------------------------------
                        ; OR   !  IR   !  CLR  !  BAI  !  DESL !  DS2  !  DS1  !  DS0  !
                        ; RO   !  RO   !  WO   !  R/W  !  R/W  !  R/W  !  R/W  !  R/W  !
                        ;-----------------------------------------------------------------

                        ;RKDC (DESIRED CYLINDER)
                        ; 15      14      13      12      11      10      9       8
                        ;-----------------------------------------------------------------
                        ; NU   !  NU   !  NU   !  NU   !  NU   !  NU   !  DC9  !  DC8  !
                        ;                          READ ONLY                            !
                        ;-----------------------------------------------------------------

                        ;  7      6       5       4       3       2       1       0
                        ;-----------------------------------------------------------------
                        ; DC7  !  DC6  !  DC5  !  DC4  !  DC3  !  DC2  !  DC1  !  DC0  !
                        ;                          READ ONLY                            !
                        ;-----------------------------------------------------------------

                        ;RKDA (DESIRED TRACK AND SECTOR)
                        ; 15      14      13      12      11      10      9       8
                        ;-----------------------------------------------------------------
                        ; NU   !  UN   !  UN   !  UN   !  UN   !  TA2  !  TA1  !  TA0  !
                        ;                          READ ONLY                            !
                        ;-----------------------------------------------------------------

                        ;  7      6       5       4       3       2       1       0
                        ;-----------------------------------------------------------------
                        ; UN   !  UN   !  UN   !  SA4  !  SA3  !  SA2  !  SA1  !  SA0  !
                        ;                          READ ONLY                            !
                        ;-----------------------------------------------------------------
```

```
2186
2187                    ;RKDB (DATA BUFFER)
2188                    ; 15     14     13     12     11     10     9      8
2189                    ;-----------------------------------------------------
2190                    ;DB15 ! DB14 ! DB13 ! DB12 ! DB11 ! DB10 ! DB9  ! DB8  !
2191                    ;                    READ/WRITE                       !
2192                    ;-----------------------------------------------------
2193
2194                    ;  7      6      5      4      3      2      1      0
2195                    ;-----------------------------------------------------
2196                    ; DB7 ! DB6  ! DB5  ! DB4  ! DB3  ! DB2  ! DB1  ! DB0  !
2197                    ;                    READ/WRITE                       !
2198                    ;-----------------------------------------------------
2199
2200                    ;RKASOF (ATTENTION SUMMARY AND OFFSET)
2201                    ; 15     14     13     12     11     10     9      8
2202                    ;-----------------------------------------------------
2203                    ;ATN7 ! ATN6 ! ATN5 ! ATN4 ! ATN3 ! ATN2 ! ATN1 ! ATN0 !
2204                    ;                    READ ONLY                        !
2205                    ;-----------------------------------------------------
2206
2207                    ;  7      6      5      4      3      2      1      0
2208                    ;-----------------------------------------------------
2209                    ; UN  ! OF6  ! OF5  ! OF4  ! OF3  ! OF2  ! OF1  ! OF0  !
2210                    ;  ?  !              READ/WRITE                       !
2211                    ;-----------------------------------------------------
2212
2213                    ;RKWC (WORD COUNT)
2214                    ; 15     14     13     12     11     10     9      8
2215                    ;-----------------------------------------------------
2216                    ;WC15 ! WC14 ! WC13 ! WC12 ! WC11 ! WC10 ! WC9  ! WC8  !
2217                    ;                    READ/WRITE                       !
2218                    ;-----------------------------------------------------
2219
2220                    ;  7      6      5      4      3      2      1      0
2221                    ;-----------------------------------------------------
2222                    ; WC7 ! WC6  ! WC5  ! WC4  ! WC3  ! WC2  ! WC1  ! WC0  !
2223                    ;                    READ/WRITE                       !
2224                    ;-----------------------------------------------------
2225
2226                    ;RKBA (BUS ADDRESS)
2227                    ; 15     14     13     12     11     10     9      8
2228                    ;-----------------------------------------------------
2229                    ;BA15 ! BA14 ! BA13 ! BA12 ! BA11 ! BA10 ! BA9  ! BA8  !
2230                    ;                    READ/WRITE                       !
2231                    ;-----------------------------------------------------
2232
2233                    ;  7      6      5      4      3      2      1      0
2234                    ;-----------------------------------------------------
2235                    ; BA7 ! BA6  ! BA5  ! BA4  ! BA3  ! BA2  ! BA1  ! BA0  !
2236                    ;                    READ/WRITE                !ALWYS0!
                        ;-----------------------------------------------------
```

```
2237                ;RKER (ERROR REGISTER)
2238                ; 15    14    13    12    11    10    9     8
2239                ;-----------------------------------------------
2240                ; DCK ! UNS ! OPI ! DTE ! WLE ! IDAE ! COE ! HVRC !
2241                ;                 READ ONLY                     !
2242                ;-----------------------------------------------
2243                ;
2244                ;  7     6     5     4     3     2     1     0
2245                ;-----------------------------------------------
2246                ; BSE ! ECH ! DTYPE! FMTE ! DRPAR! ILF ! SKI ! ILC !
2247                ;                 READ ONLY                     !
2248                ;-----------------------------------------------
2249                ;
2250                ;RKDS (DRIVE STATUS)
2251                ; 15    14    13    12    11    10    9     8
2252                ;-----------------------------------------------
2253                ;SVAL ! DSC ! PIP ! SPON ! WRL ! UN ! UN ! DTP !
2254                ;                 READ ONLY                     !
2255                ;-----------------------------------------------
2256                ;
2257                ;  7     6     5     4     3     2     1     0
2258                ;-----------------------------------------------
2259                ;DRDY !  VV ! DROT ! SPLS ! ACLO ! OFSET!  UN ! DRA !
2260                ;                 READ ONLY                     !
2261                ;-----------------------------------------------
2262                ;
2263                ;RKMR1 (MAINTENANCE REG 1)
2264                ; 15    14     13     12     11     10     9      8
2265                ;-----------------------------------------------------
2266                ;RD    ! WRT  ! ECCW ! PCD  ! PCA  ! MEWD ! MERD ! MCLK !
2267                ;GATE ! GATE !      !      !      !      !      !      !
2268                ;                 READ ONLY                 ! READ/WRITE !
2269                ;-----------------------------------------------------
2270                ;
2271                ;  7      6      5      4      3      2      1      0
2272                ;-----------------------------------------------------
2273                ;MIND ! MSP  ! DMD  ! PAT  ! MS3  ! MS2  ! MS1  ! MS0  !
2274                ;                 READ/WRITE                          !
2275                ;-----------------------------------------------------
```

```
2276                ;RKECC/PAT
2277                ;  15    14    13    12    11    10    9     8
2278                ;-------------------------------------------------
2279                ;  UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
2280                ;                   READ ONLY                     !
2281                ;-------------------------------------------------
2282
2283                ;   7     6     5     4     3     2     1     0
2284                ;-------------------------------------------------
2285                ; EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
2286                ;                   READ ONLY                     !
2287                ;-------------------------------------------------
2288
2289                ;RKECC/POS
2290                ;  15    14    13    12    11    10    9     8
2291                ;-------------------------------------------------
2292                ;  UN  !  UN  !  UN  ! EP012! EP011! EP010! EP09 ! EP08 !
2293                ;                   READ ONLY                     !
2294                ;-------------------------------------------------
2295
2296                ;   7     6     5     4     3     2     1     0
2297                ;-------------------------------------------------
2298                ; EP07 ! EP06 ! EP05 ! EP04 ! EP03 ! EP02 ! EP01 ! EP00 !
2299                ;                   READ ONLY                     !
2300                ;-------------------------------------------------
2301
2302                ;DRIVE STATUS INFORMATION
2303
2304                ;LINE A MESSAGE 00
2305                ;  15    14    13    12    11    10    9     8
2306                ;-------------------------------------------------
2307                ;  PAR ! DSC  ! PIP  ! SPON ! WRLK ! OFFON! FMT  ! DRTP !
2308                ;-------------------------------------------------
2309
2310                ;   7     6     5     4     3     2     1     0
2311                ;-------------------------------------------------
2312                ; DRDY !  VV  ! DRA  !  NU  !  NU  ! DRIVE SELECT CODE  !
2313                ;-------------------------------------------------
```

```
;LINE B MESSAGE 00
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; PAR ! UNS  ! DROT ! DCLO ! WLE  ! SKI  ! PERR ! ILF  !
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
; FLT ! ACLO ! IVDA !  UN  !  UN  !  UN  !  0   !  0   !
;-----------------------------------------------------

;LINE A MESSAGE 01
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; PAR ! UNLDG! RTZ  ! LDG  ! REV  ! FWD  ! SPEED! CART !
;     ! HDS  !      ! HDS  !      !      !  OK  ! PRES !
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
;DOOR ! BRUSH! HEADS! TRK  !  UN  !  DRIVE SELECT CODE !
;LTCH ! HOME ! HOME ! FOLOW!      !                     !
;     !      !      !  OK  !      !                     !
;-----------------------------------------------------

;LINE B MESSAGE 01
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; PAR ! SERVO! LIMIT! SEEK ! PLO  ! DIBIT! INDEX! MULTI!
;     ! BRAKE! ON SK!NO MO ! ERR  !  ERR !  ERR !HD SEL!
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
;HEAD !WR GTE!WR CUR! SEC  !  NU  !  NU  !  0   !  1   !
;FAULT!AND NO!AND NO! ERR  !      !      !      !      !
;     !XISTON!WR GTE!      !      !      !      !      !
;-----------------------------------------------------
```

```
                        ;LINE A MESSAGE 10
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ; PAR !  NU  !  NU  ! CYLINDER DIFF/OFFSET VALUE      !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ;CYLINDER DIFF/OFFSET VALUE!  NU  !  DRIVE SELECT CODE !
                        ;-----------------------------------------------------
                        ;
                        ;LINE B MESSAGE 10
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ; PAR ! ALIGN!  NU  !    CYLINDER ADDRESS             !
                        ;     ! SIGN !                                        !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ;    CYLINDER ADDRESS      ! UN ! UN !  1 !  0 !
                        ;-----------------------------------------------------
                        ;
                        ;LINE A MESSAGE 11
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ; PAR !     DRIVE SERIAL NUMBER                      !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ; DRIVE SERIAL NUMBER            !  DRIVE SELECT CODE !
                        ;-----------------------------------------------------
                        ;
                        ;LINE B MESSAGE 11
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ; PAR !  NU  !    NU     !   DECODED HEAD     ! SECTOR!
                        ;     !     !           !     ADDRESS         ! COUNT !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ;   SECTOR COUNT          ! UN ! UN !  1 !  1 !
                        ;-----------------------------------------------------
                        ;

                        .SBTTL  RK06 CONTROLLER REGISTER DEFINITION

000000                  RKCS1=  0               ;CONTROL AND STATUS REGISTER 1
000002                  RKWC=   2               ;WORD COUNT REGISTER
000004                  RKBA=   4               ;BUS ADDRESS REGISTER
000006                  RKDA=   6               ;DESIRED TRACK SECTOR REGISTER
000010                  RKCS2=  10              ;CONTROL AND STATUS REGISTER 2
```

```
                        RKDS=   12                      ;DRIVE STATUS REGISTER
                        RKER=   14                      ;ERROR REGISTER
                        RKASOF= 16                      ;ATTENTION SUMMARY AND OFFSET REGISTER
                        RKDC=   20                      ;DESIRED CYLINDER REGISTER
                        RKCYL=  20                      ;DESIRED CYLINDER REGISTER
                        RKDB=   24                      ;DATA BUFFER
                        RKMR1=  26                      ;MAINTENANCE REGISTER 1
                        RKMR2=  34                      ;MAINTENANCE REGISTER 2
                        RKMR3=  36                      ;MAINTENANCE REGISTER 3
                        RKPOS=  30                      ;ECC POSITION INFORMATION
                        RKECPS= 30                      ;ECC POSITION INFORMATION
                        RKPAT=  32                      ;ECC PATTERN INFORMATION
                        RKECPT= 32                      ;ECC PATTERN INFORMATION

                        .SBTTL  DRIVE COMMANDS

                        SELDRV= 101                     ;SELECT DRIVE
                        PACK=   103                     ;PACK ACKNOWLEDGE
                        CLEAR=  105                     ;DRIVE CLEAR
                        UNLOAD= 107                     ;UNLOAD
                        SRTSPL= 111                     ;START SPINDLE
                        RECAL=  113                     ;RECALIBRATE
                        OFFSET= 115                     ;OFFSET
                        SEEK=   117                     ;SEEK
                        RDDATA= 121                     ;READ DATA
                        WRDATA= 123                     ;WRITE DATA
                        RDHEAD= 125                     ;READ HEADER
                        WRHEAD= 127                     ;WRITE HEADER AND DATA
                        WRTCHK= 131                     ;WRITE CHECK

                        ;       THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
                        ;       TO SIMULATE A SPECIFIC DESIRED OPERATION

                        RELEAS= 140                     ;RELEASE DRIVE
                        RDSTAT= 141                     ;GET ALL STATUS FROM DRIVE
                        RDALHD= 164                     ;READ ALL HEADERS
                        CONCLR= 176                     ;CONTROLLER CLEAR (BIT 15 OF CS1)
                        SUBCLR= 177                     ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
                        INTR=   300                     ;GENERATE INTERRUPT TO CPU

                        ;       DRIVER ISSUED SERVICE COMMANDS

                        DR.SEL= 001                     ;DRIVE SELECT
                        DR.CLR= 005                     ;DRIVE CLEAR

                        .SBTTL  CONTROL AND STATUS REGISTER 1 BITS

                        GO=     BIT0                    ;GO BIT
                        IE=     BIT6                    ;INTERRUPT ENABLE
                        RDY=    BIT7                    ;CONTROLLER READY
                        BA16=   BIT8                    ;BUS ADDRESS BIT 16
                        BA17=   BIT9                    ;BUS ADDRESS BIT 17
                        CDT=    BIT10                   ;CONTROLLER DRIVE TYPE (0=RK06)
                        CTO=    BIT11                   ;CONTROLLER TIMED OUT WAITING FOR
                                                        ;  DRIVE RESPONSE
                        CFMT=   BIT12                   ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
```

```
2464            020000          SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
2465            040000          DI=     BIT14                   ;DRIVE INTERRUPT
2466            100000          CERR=   BIT15                   ;CONTROLLER ERROR
2467            100000          CCLR=   BIT15                   ;CONTROLLER CLEAR
2468
2469                            ;       THESE BIT DEFINITIONS ARE USED FOR ADDRESS
2470                            ;       THE HIGH BYTE OF RKCS1
2471
2472            000001          B.BA16= BIT0                    ;BUS ADDRESS BIT 16
2473            000002          B.BA17= BIT1                    ;BUS ADDRESS BIT 17
2474            000004          B.CDT=  BIT2                    ;CONTROLLER DRIVE TYPE (0=RKD6)
2475            000020          B.CFMT= BIT4                    ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
2476
2477                            .SBTTL  CONTROL AND STATUS REGISTER 2 BITS
2478
2479            000007          DRVMSK= 7                       ;MASK FOR DRIVE SELECTION CODE
2480            000010          DESL=   BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
2481            000010          RLS=    BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
2482            000020          BAI=    BIT4                    ;BUS ADDRESS INCREMENT INHIBIT
2483            000040          CLR=    BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
2484            000040          SCLR=   BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
2485            000100          IR=     BIT6                    ;INPUT READY
2486            000200          OR=     BIT7                    ;OUTPUT READY
2487            000400          UFE=    BIT8                    ;UNIT FIELD ERROR
2488            001000          MDS=    BIT9                    ;MULTIPLE DRIVE SELECT
2489            002000          PGE=    BIT10                   ;PROGRAMMING ERROR
2490            004000          NEM=    BIT11                   ;NON-EXISTENT MEMORY
2491            010000          NED=    BIT12                   ;NON-EXISTENT DRIVE
2492            020000          UPE=    BIT13                   ;UNIBUS PARITY ERROR
2493            040000          WCE=    BIT14                   ;WRITE CHECK ERROR
2494            100000          DLT=    BIT15                   ;DATA LATE ERROR
2495
2496                            .SBTTL  ERROR REGISTER BIT DEFINITION
2497
2498            000001          ILC=    BIT0                    ;ILLEGAL FUNCTION CODE
2499                            ;*ILF=  BIT0                    ;ILLEGAL FUNCTION CODE
2500            000002          SKI=    BIT1                    ;SEEK INCOMPLETE
2501            000004          ILF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
2502            000004          NXF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
2503            000010          DRPAR=  BIT3                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
2504            000020          FMTE=   BIT4                    ;FORMAT ERROR
2505            000040          DTYE=   BIT5                    ;DRIVE TYPE ERROR
2506            000100          ECH=    BIT6                    ;ECC HARD
2507            000200          BSE=    BIT7                    ;BAD SECTOR ERROR
2508            000400          HCRC=   BIT8                    ;HEADER CRC ERROR
2509            000400          HVRC=   BIT8                    ;HEADER VRC ERROR
2510            001000          COE=    BIT9                    ;CYLINDER ADDRESS OVERFLOW ERROR
2511            002000          IDAE=   BIT10                   ;INVALID DISK ADDRESS ERROR
2512            004000          WLE=    BIT11                   ;WRITE LOCK ERROR
2513            010000          DTE=    BIT12                   ;DRIVE TIMING ERROR
2514            020000          OPI=    BIT13                   ;OPERATION (SEARCH) INCOMPLETE
2515            040000          UNS=    BIT14                   ;DRIVE UNSAFE
2516            100000          DCK=    BIT15                   ;DATA CHECK
2517
2518                            .SBTTL  STATUS REGISTER BIT DEFINITION
2519
```

```
2520            000001          DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
2521                                                            ;   THIS BIT IS RESET)
2522            000004          OFST=   BIT2                    ;DRIVE OFFSET
2523            000010          ACLO=   BIT3                    ;AC LOW
2524            000020          SPDLSS= BIT4                    ;SPEED LOSS
2525            000020          DCLO=   BIT4                    ;DC LOW
2526            000040          DROT=   BIT5                    ;DRIVE OFF TRACK
2527            000100          VV=     BIT6                    ;VOLUME VALID
2528            000200          DRY=    BIT7                    ;DRIVE READY
2529            000200          DRDY=   BIT7                    ;DRIVE READY
2530            000400          DDT=    BIT8                    ;DRIVE TYPE (0=RK06)
2531            004000          WRL=    BIT11                   ;WRITE LOCK
2532            020000          PIP=    BIT13                   ;POSITIONING IN PROGRESS
2533            040000          DSC=    BIT14                   ;DRIVE STATUS CHANGE
2534            100000          SVAL=   BIT15                   ;STATUS VALID
2535
2536                            .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
2537
2538            000017          MESMSK= 17                      ;MESSAGE MASK
2539
2540            000020          PAT=    BIT4                    ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
2541            000040          DMD=    BIT5                    ;DIAGNOSTIC MODE
2542            000100          MSP=    BIT6                    ;MAINTENANCE SECTOR PULSE
2543            000200          MIND=   BIT7                    ;MAINTENANCE INDEX
2544            000400          MCLK=   BIT8                    ;MAINTENANCE CLOCK
2545            001000          MERD=   BIT9                    ;MAINTENANCE ENCODED READ DATA
2546            002000          MEWD=   BIT10                   ;MAINTENACNE ENCODED WRITE DATA
2547            004000          PCA=    BIT11                   ;PRECOMPENSATION ADVANCE
2548            010000          PCD=    BIT12                   ;PRECOMPENSATION DELAY
2549            020000          ECCW=   BIT13                   ;ECC WORD IS BEING READ OR WRITTEN
2550            040000          WRTGAT= BIT14                   ;WRITE GATE
2551            100000          RDGATE= BIT15                   ;READ GATE
2552
2553                            .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
2554
2555            000040          S.DRA=  BIT5                    ;DRIVE AVAILIABLE
2556            000100          S.VV=   BIT6                    ;VOLUME VALID
2557            000200          S.DRY=  BIT7                    ;DRIVE READY
2558            000400          S.TYPE= BIT8                    ;DRIVE TYPE
2559            001000          S.FORM= BIT9                    ;DRIVE FORMAT
2560            002000          S.OFF=  BIT10                   ;OFFSET
2561            004000          S.WRL=  BIT11                   ;WRITE LOCK
2562            010000          S.SPIN= BIT12                   ;SPINDLE ON
2563            020000          S.PIP=  BIT13                   ;POSITIONING IN PROGRESS
2564            040000          S.DSC=  BIT14                   ;DRIVE STATUS CHANGE
2565
2566                            .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
2567
2568            000040          S.ICYL= BIT5                    ;ILLEGAL CYLINDER ADDRESS
2569            000100          S.ACLO= BIT6                    ;AC LOW
2570            000200          S.FLT=  BIT7                    ;DRIVE FAULT
2571            000400          S.ILF=  BIT8                    ;ILLEGAL FUNCTION
2572            001000          S.PAR=  BIT9                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
2573            002000          S.SKI=  BIT10                   ;SEEK INCOMPLETE
2574            004000          S.WLE=  BIT11                   ;WRITE LOCK ERROR
2575            010000          S.SPLS= BIT12                   ;SPEED LOSS
```

```
                010000          S.DCLO= BIT12                   ;DC LOW
                020000          S.DROT= BIT13                   ;DRIVE OFF TRACK
                040000          S.UNS=  BIT14                   ;DRIVE UNSAFE

                                .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A

                000020          S.XDOK= BIT4                    ;TRANSDUCER OK
                000040          S.HDHM= BIT5                    ;HEADS HOME
                000100          S.BRHM= BIT6                    ;BRUSHES HOME
                000200          S.DOOR= BIT7                    ;DOOR INTERLOCKED
                000400          S.CART= BIT8                    ;CARTRAGE INTERLOCK
                001000          S.SPOK= BIT9                    ;SPEED OK
                002000          S.FWD=  BIT10                   ;FORWARD
                004000          S.REV=  BIT11                   ;REVERSE
                010000          S.LOAD= BIT12                   ;HEADS LOADING
                020000          S.RTZ=  BIT13                   ;RETURN TO ZERO
                040000          S.UNLD= BIT14                   ;HEADS UNLOADING

                                .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B

                000020          S.SECT= BIT4                    ;SECTOR ERROR
                000040          S.WCLK= BIT5                    ;WRITE CLOCK AND NO WRITE GATE
                000100          S.WGAT= BIT6                    ;WRITE GATE AND NO TRANSISTIONS
                000200          S.HDFL= BIT7                    ;HEAD FAULT
                000400          S.MHD=  BIT8                    ;MULTIPLE HEAD SELECT
                001000          S.XERR= BIT9                    ;INDEX ERROR
                002000          S.DIB=  BIT10                   ;DIBIT ERROR
                004000          S.PLO=  BIT11                   ;PLO ERROR
                010000          S.NMOV= BIT12                   ;SEEK AND NO MOTION
                020000          S.LIMD= BIT13                   ;LIMIT DETECT ON SEEK
                040000          S.BRKE= BIT14                   ;SERVO-BRAKE

                                .SBTTL  COMMON MASKS

                000007          M.DRV=  7                       ;DRIVE CODE
                100000          M.PAR=  BIT15                   ;PARITY
                000003          M.ID=   3                       ;BYTE ID
                017760          M.CDIF= 17760                   ;CYLINDER DIFFERENCE/OFFSET
                017760          M.CADD= 17760                   ;CYLINDER ADDRESS
                077770          M.SER=  77770                   ;DRIVE SERIAL NUMBER
                000760          M.SECT= 760                     ;SECTOR COUNT
                007000          M.HEAD= 7000                    ;HEAD DECODE
```

```
                        .SBTTL  PARAMETER BLOCK ALLOCATION

                        ;*      ************************************************
                        ;*    1 ! COMMAND                    ! DRIVE NO.       !  0
                        ;*    3 ! CYLINDER ADDRESS                             !  2
                        ;*    5 ! TRACK                      ! SECTOR          !  4
                        ;*    7 !BA16-17,FORMAT,DRV TYPE     ! OFFSET          !  6
                        ;*   11 ! BUS ADDRESS (LOW 16 BITS)                    ! 10
                        ;*   13 ! WORD COUNT (2'S COMPLEMENT)                  ! 12
                        ;*   15 ! PROGRAM DRIVE STATUS INFORMATION             ! 14
                        ;*   17 ! COMMAND AND STATUS REGISTER 1                ! 16
                        ;*   21 ! COMMAND AND STATUS REGISTER 2                ! 20
                        ;*   23 ! WORD COUNT REGISTER                          ! 22
                        ;*   25 ! BUS ADDRESS REGISTER                         ! 24
                        ;*   27 ! DESIRED TRACK AND SECTOR                     ! 26
                        ;*   31 ! DESIRED CYLINDER                             ! 30
                        ;*   33 ! ATTENTION SUMMARY AND DRIVE OFFSET           ! 32
                        ;*   35 ! ERROR REGISTER                               ! 34
                        ;*   37 ! STATUS REGISTER                              ! 36
                        ;*   41 ! MESSAGE LINE A STATUS BYTE 00                ! 40
                        ;*   43 ! MESSAGE LINE B STATUS BYTE 00                ! 42
                        ;*   45 ! MESSAGE LINE A STATUS BYTE 01                ! 44
                        ;*   47 ! MESSAGE LINE B STATUS BYTE 01                ! 46
                        ;*   51 ! MESSAGE LINE A STATUS BYTE 10                ! 50
                        ;*   53 ! MESSAGE LINE B STATUS BYTE 10                ! 52
                        ;*   55 ! MESSAGE LINE A STATUS BYTE 11                ! 54
                        ;*   57 ! MESSAGE LINE B STATUS BYTE 11                ! 56
                        ;*   61 ! ECC POSITION INFORMATION                     ! 60
                        ;*   63 ! ECC PATTERN INFORMATION                      ! 62
                        ;*      ************************************************

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

                        ;       THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
                        ;         TO THE RK06 DRIVER

000000                  P.DRVN= 0                       ;DRIVE NUMBER
000001                  P.CMND= 1                       ;COMMAND
000002                  P.CYLN= 2                       ;CYLINDER ADDRESS
000004                  P.SECT= 4                       ;SECTOR
000005                  P.TRCK= 5                       ;TRACK
000006                  P.OFST= 6                       ;OFFSET
000007                  P.CS1H= 7                       ;RKCS1 BITS 8-15
000007                  P.BAHI= 7                       ;BUS ADDRESS (BITS 16 AND 17)
000010                  P.BALO= 10                      ;BUS ADDRESS (BITS 0-15)
000012                  P.WC=   12                      ;WORD COUNT (2'S COMPLEMENT)
000014                  P.PRST= 14                      ;PROGRAM DRIVE STATUS INFORMATION

                        .SBTTL  PROGRAM DEVICE STATUS REGISTER DEFINITION

000001                  DRVUSE= BIT0                    ;DRIVE IN USE
000002                  DRVPOS= BIT1                    ;DRIVE POSITIONING
000004                  DRVPDT= BIT2                    ;DRIVE POSITIONED FOR DATA TRANSFER
000010                  UEXATT= BIT3                    ;UNEXPECTED ATTENTION
000020                  DRVHRD= BIT4                    ;DRIVE HAS HARD ERROR
000040                  DRVDSC= BIT5                    ;DRIVE STATUS CHANGE DID NOT CLEAR
```

```
                    000100          CMDTO=  BIT6                    ;NO TERMINATION TO COMMAND FOR AT
                                                                    ;  LEAST 1 SECOND
                    000200          W.WCK=  BIT7                    ;WRITE FOR WRITE WRITE CHECK
                    000400          NOCHK=  BIT8                    ;NO CHECK, DO NOT SET INTERRUPT ENABLE
                    001000          PBSVAL= BIT9                    ;PARAMETER STATUS WORDS VALID
                                                                    ;  (SET WHEN ERROR TERMINATION OR
                                                                    ;  READ STATUS COMMAND)
                    002000          DRPDRV= BIT10                   ;DROP DRIVE FROM TEST SEQUENCE
                    004000          NODSC=  BIT11                   ;ATTENTION SET BUT DCS AND FAULT RESET
                    010000          DRVSZD= BIT12                   ;DRIVE SEIZED BY OTHER PORT
                    020000          E.UNLD= BIT13                   ;DRIVE UNLOADED DUE TO ERROR
                    040000          Q.INIT= BIT14                   ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
                    100000          DTBAII= BIT15                   ;INHIBIT BUS ADDRESS INCREMENT

                                    .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM

                                    ;       THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
                                    ;         FROM THE DRIVER TO THE CALLING PROGRAM

                    000016          P.CS1=  16                      ;COMMAND AND STATUS REGISTER 1
                    000020          P.CS2=  20                      ;COMMAND AND STATUS REGISTER 2
                    000022          P.WCR=  22                      ;WORD COUNT REGISTER
                    000024          P.BAR=  24                      ;BUS ADDRESS REGISTER
                    000026          P.DTS=  26                      ;DESIRED TRACK SECTOR REGISTER
                    000030          P.DCYL= 30                      ;DESIRED CYLINDER REGISTER
                    000032          P.ASOF= 32                      ;ATTENTION SUMMARY/OFFSET REGISTER
                    000034          P.ER=   34                      ;ERROR REGISTER
                    000036          P.DS=   36                      ;STATUS REGISTER
                    000040          P.A00=  40                      ;MESSAGE A STATUS BYTE 00
                    000042          P.B00=  42                      ;MESSAGE B STATUS BYTE 00
                    000044          P.A01=  44                      ;MESSAGE A STATUS BYTE 01
                    000046          P.B01=  46                      ;MESSAGE B STATUS BYTE 01
                    000050          P.A10=  50                      ;MESSGGE A STATUS BYTE 10
                    000052          P.B10=  52                      ;MESSAGE B STATUS BYTE 10
                    000054          P.A11=  54                      ;MESSAGE A STATUS BYTE 11
                    000056          P.B11=  56                      ;MESSAGE B STATUS BYTE 11
                    000060          P.EPOS= 60                      ;ECC POSITION INFORMATION
                    000062          P.EPAT= 62                      ;ECC PATTERN INFORMATION

                                    .SBTTL  PARAMETER BLOCK 0 FOR DRIVE

002630      000     PARM0:  .BYTE   0                               ;DRIVE NUMBER
002631      000             .BYTE   0                               ;COMMAND
002632  000000              .WORD   0                               ;CYLINDER ADDRESS
002634      000             .BYTE   0                               ;SECTOR ADDRESS
002635      000             .BYTE   0                               ;TRACK ADDRESS
002636      000             .BYTE   0                               ;OFFSET VALUE
002637      000             .BYTE   0                               ;BUS ADDRESS (BITS 16 AND 17)
002640  000000              .WORD   0                               ;BUS ADDRESS (BITS 0 - 15)
002642  000000              .WORD   0                               ;WORD COUNT (2'S COMPLEMENT)
002644  000000              .WORD   0                               ;PROGRAM DRIVE STATUS INFORMATION
002646  000000              .WORD   0                               ;COMMAND AND STATUS REGISTER 1
002650  000000              .WORD   0                               ;COMMAND AND STATUS REGISTER 2
002652  000000              .WORD   0                               ;WORD COUNT REGISTER
002654  000000              .WORD   0                               ;BUS ADDRESS REGISTER
002656  000000              .WORD   0                               ;DESIRED TRACK AND SECTOR REGISTER
```

```
2730  002660  000000                    .WORD   0       ;DESIRED CYLINDER REGISTER
2731  002662  000000                    .WORD   0       ;ATTENTION SUMMARY/OFFSET REGISTER
2732  002664  000000                    .WORD   0       ;ERROR REGISTER
2733  002666  000000                    .WORD   0       ;STATUS REGISTER
2734  002670  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 00
2735  002672  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 00
2736  002674  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 01
2737  002676  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 01
2738  002700  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 10
2739  002702  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 10
2740  002704  000000                    .WORD   0       ;MESSAGE LINE A STATUS BYTE 11
2741  002706  000000                    .WORD   0       ;MESSAGE LINE B STATUS BYTE 11
2742  002710  000000                    .WORD   0       ;ECC POSITION INFORMATION
2743  002712  000000                    .WORD   0       ;ECC PATTERN INFORMATION
2744
2745                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
2746
2747  002714      000         PARM1:  .BYTE   0       ;DRIVE NUMBER
2748  002715      000                 .BYTE   0       ;COMMAND
2749  002716  000000                  .WORD   0       ;CYLINDER ADDRESS
2750  002720      000                 .BYTE   0       ;SECTOR ADDRESS
2751  002721      000                 .BYTE   0       ;TRACK ADDRESS
2752  002722      000                 .BYTE   0       ;OFFSET VALUE
2753  002723      000                 .BYTE   0       ;BUS ADDRESS (BITS 16 AND 17)
2754  002724  000000                  .WORD   0       ;BUS ADDRESS (BITS 0 - 15)
2755  002726  000000                  .WORD   0       ;WORD COUNT (2'S COMPLEMENT)
2756  002730  000000                  .WORD   0       ;PROGRAM DRIVE STATUS INFORMATION
2757  002732  000000                  .WORD   0       ;COMMAND AND STATUS REGISTER 1
2758  002734  000000                  .WORD   0       ;COMMAND AND STATUS REGISTER 2
2759  002736  000000                  .WORD   0       ;WORD COUNT REGISTER
2760  002740  000000                  .WORD   0       ;BUS ADDRESS REGISTER
2761  002742  000000                  .WORD   0       ;DESIRED TRACK AND SECTOR REGISTER
2762  002744  000000                  .WORD   0       ;DESIRED CYLINDER REGISTER
2763  002746  000000                  .WORD   0       ;ATTENTION SUMMARY/OFFSET REGISTER
2764  002750  000000                  .WORD   0       ;ERROR REGISTER
2765  002752  000000                  .WORD   0       ;STATUS REGISTER
2766  002754  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 00
2767  002756  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 00
2768  002760  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 01
2769  002762  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 01
2770  002764  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 10
2771  002766  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 10
2772  002770  000000                  .WORD   0       ;MESSAGE LINE A STATUS BYTE 11
2773  002772  000000                  .WORD   0       ;MESSAGE LINE B STATUS BYTE 11
2774  002774  000000                  .WORD   0       ;ECC POSITION INFORMATION
2775  002776  000000                  .WORD   0       ;ECC PATTERN INFORMATION
2776
2777                          .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
2778
2779  003000  000000          T.CS1:  .WORD   0       ;TEMPORARY STORAGE FOR COMMAND AND STATUS
2780                                                  ;  REGISTER 1
2781  003002  000000          T.CS2:  .WORD   0       ;TEMPORARY STORAGE FOR COMMAND AND STATUS
2782                                                  ;  REGISTER 2
2783  003004  000000          T.WCR:  .WORD   0       ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
2784  003006  000000          T.BA:   .WORD   0       ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
2785  003010  000000          T.DA:   .WORD   0       ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
2786  003012  000000          T.DC:   .WORD   0       ;TEMPORARY STORAGE FOR DRIVE CYLINDER
2787  003014  000000          T.ASOF: .WORD   0       ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
2788                                                  ;  AND OFFSET
2789  003016  000000          T.ER:   .WORD   0       ;TEMPORARY STORAGE FOR ERROR REGISTER
2790  003020  000000          T.DS:   .WORD   0       ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
2791  003022  000000          T.MR1:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
2792  003024  000000          T.MR2:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
2793  003026  000000          T.MR3:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
2794  003030  000000          T.POS:  .WORD   0       ;TEMPORARY STORAGE FOR ECC POSITION
2795  003032  000000          T.PAT:  .WORD   0       ;TEMPORARY STORAGE FOR ECC PATTERN
2796  003034  000000          T.DB:   .WORD   0       ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
2797
2798                          .SBTTL  DRIVER PARAMERTERS
2799
2800  003036  177440          RKBAS:  .WORD   177440  ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
2801  003040  000210          RKVEC:  .WORD   210     ;ADDRESS OF R611 VECTOR
2802  003042  000240          RKPRI:  .WORD   PR5     ;RK611 INTERRUPT PRIORITY
2803  003044  027426          A.NORM: ERRFRE          ;ADDRESS OF NORMAL RETURN FROM DRIVER
2804  003046  030640          A.ABNL: ERRHDL          ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
2805  003050  030134          A.CONT: CONERR          ;ADDRESS OF CONTROLLER ERROR RETURN
2806  003052  000000          E.CONT: .WORD   0       ;CONTROLLER ERROR STATUS
2807                                                  ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
2808                                                  ;  IS INITIATED.  IF A CONTROLLER ERROR
2809                                                  ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
2810                                                  ;  USED:
2811                                                  ;
2812          000001          E.CCLR= BIT0            ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2813          000002          E.NOAT= BIT1            ;NO ATTENTION IN ATTENTION SUMMARY REG
2814          000004          E.UATT= BIT2            ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
2815          000010          E.UDAT= BIT3            ;UNEXPECTED DATA TYPE ERROR
2816          000020          E.CLAT= BIT4            ;ATTENTION DID NOT RESET WITH CLEAR
2817          000040          E.SCLR= BIT5            ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
2818                                                  ;  ATTENTION
2819          000100          E.ILLD= BIT6            ;ILLEGAL DRIVER COMMAND
2820          000400          E.DLT=  BIT8            ;DATA LATE WHEN UNLOADING HEADER
2821          001000          E.CERR= BIT9            ;CONTROLLER ERROR DURING DRIVER SERVICING
2822          002000          E.DPAR= BIT10           ;DRIVE DETECTED PARITY ERROR
2823          040000          E.CMTO= BIT14           ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
2824          100000          E.MDS=  BIT15           ;MULTIPLE DRIVE SELECT
2825
2826  003054  000000          O.WAIT: .WORD   0       ;PARAMETER BLOCK OF THE DRIVE
2827                                                  ;WAITING FOR COMMAND COMPLETION
2828  003056  000400          W.MTIM: .WORD   400     ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
2829  003060  000400          W.MILI: .WORD   400     ;16 MILISECOND TIME FOR PROGRAM
2830
2831                                                  ;       CPU     VALUE
2832                                                  ;       ---     -----
2833
2834                                                  ;       11/05   100
2835                                                  ;       11/10
2836                                                  ;       11/20
2837                                                  ;       11/34
2838                                                  ;       11/40
2839                                                  ;       11/45   400
2840                                                  ;       11/50
2841                                                  ;       11/70
```

```
2842  003062  000300                  W.SEC:  .WORD   300             ;SECOND COUNT COUNT FOR ALL COMMANDS
2843                                                                  ;  EXCEPT START SPINDLE
2844  003064  003000                  W.BSEC: .WORD   3000            ;8 SECOND FOR DRIVE CYCLE DOWN
2845  003066  030000                  W.MIN:  .WORD   30000           ;MINUTE TIME FOR START SPINDLE
2846  003070  000000                  HDR.AD: .WORD   0               ;ADDRESS USED FOR READ ALL HEADERS
2847  003072  000000                  HDR.CT: .WORD   0               ;NUMBER OF HEADERS LEFT TO READ FOR READ
2848                                                                  ;  ALL HEADERS
2849  003074      000                 I.ISRL: .BYTE   0               ;INTERRUPT OR RELEASED COMMAND ISSUED
2850  003075      002     004     010 H.HEAD: .BYTE   2,4,10          ;HEAD DECODES
2851  003100      000                 W.TIME: .BYTE   0               ;DRIVES BEING WATCH-DOG TIMED
2852
2853                                  .SBTTL  INTERRUPT MASKS
2854
2855  003101      000                 INTMSK: .BYTE   0               ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
2856
2857                                  ;       INTERRUPT MASK TABLE
2858
2859  003102      001                 I.DRV:  .BYTE   1               ;INTERRUPT MASK FOR DRIVE 0
2860  003103      002                         .BYTE   2               ;INTERRUPT MASK FOR DRIVE 1
2861  003104      004                         .BYTE   4               ;INTERRUPT MASK FOR DRIVE 2
2862  003105      010                         .BYTE   10              ;INTERRUPT MASK FOR DRIVE 3
2863  003106      020                         .BYTE   20              ;INTERRUPT MASK FOR DRIVE 4
2864  003107      040                         .BYTE   40              ;INTERRUPT MASK FOR DRIVE 5
2865  003110      100                         .BYTE   100             ;INTERRUPT MASK FOR DRIVE 6
2866  003111      200                         .BYTE   200             ;INTERRUPT MASK FOR DRIVE 7
2867
2868                                  .SBTTL  PARAMETER BLOCK TABLE
2869
2870  003112  002630                  PBLKT:  PARM0                   ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
2871                                                                  ;DRIVE CALL.  MUST BE LOADED INTO PBLKT
2872
2873
2874                                  .SBTTL  TIME FOR WATCH-DOG TIMER
2875
2876  003114  000000                  W.DRV:  .WORD   0               ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
2877                                  .SBTTL  PROGRAM SPECIFIC RESERVED LOCATIONS
2878  003116      000                 MDFLAG: .BYTE   0               ;FLAG TO INDIC. DEFLT OR PARAM MODE
2879  003117      000                 TSTING: .BYTE   0               ;CURRENTLY RUNNING TESTS IF = 1
2880  003120      000                 DERCNT: .BYTE   0               ;DATA ERROR COUNT
2881  003121      000                 OPCOMP: .BYTE   0               ;OPERATION COMPLETE FLAG
2882  003122      000                 DONE:   .BYTE   0               ;DONE SWITCH
2883  003123      000                 TYPFMT: .BYTE   0               ;DRIVE TYPE & FORMAT CONTROL
2884  003124      000                 FORMAT: .BYTE   0               ;DRIVE FORMAT IN BIT 4 OF BYTE
2885  003125      000                 ERRCNT: .BYTE   0               ;ERROR COUNT
2886  003126      004                 ERRLMT: .BYTE   4               ;ERROR LIMIT
2887  003127      000                 DRVERS: .BYTE   0               ;ERROR COUNT FOR CURRENT DRIVE
2888  003130      000                 OPCONT: .BYTE   0               ;OPERATION CONTROL SWITCHES
2889  003131      000                 DRNAFG: .BYTE   0               ;=1 INDICATES DRIVE SIEZED BY OTHER PORT
2890  003132      000                 REISSU: .BYTE   0               ;DUAL-ACC FLAG TO RE-ISSUE COMMAND
2891  003133      000                 TSTTYP: .BYTE   0               ;OFFSET TEST IDENTIFIER
2892  003134      000                 DCEFLG: .BYTE   0               ;DATA CHECK ERROR FLAG
2893  003135      000                 WCEFLG: .BYTE   0               ;WRITE CHECK ERROR FLAG
2894  003136      000                 DLTFLG: .BYTE   0               ;DATA LATE ERROR FLAG
2895  003137      000                 DIF100: .BYTE   0               ;"CYL DIF = 100" FLAG
2896  003140      000                 NORTRY: .BYTE   0               ;"NO-RETRY" FLAG
2897  003141      000                 MEMABT: .BYTE   0               ;SET BYTE = 1 FOR NO ABORT
```

```
2898                                                    ;     ON MEMORY PARITY ERRORS
2899  003142  000                 HLPOVL: .BYTE   0     ;HELP FILE OVERLAID INDICATOR
2900  003143  000                 NOTYPE: .BYTE   0     ;INDICATES PROGRAM JUST LOADED IF 0
2901  003144  000                 LOGDRV: .BYTE   0     ;LOGICAL DRIVE NUMBER (00-17)
2902  003145  000                 PASSNO: .BYTE   0     ;PASS NUMBER (= 1 OR 2)
2903  003146  000                 OFSDIR: .BYTE   0     ;OFFSET DIRECTION FLAG (0=NEG)
2904  003147  000                 OFSCNT: .BYTE   0     ;OFFSET COUNT FOR ONE DIRECTION
2905          000001              WHDSW=BIT0            ;WRITE HEADER & DATA SWITCH
2906          000002              VFHDSW=BIT1           ;VERIFY HEADERS SWITCH
2907          000004              WCDASW=BIT2           ;WRITE CHECK DATA SWITCH
2908          000010              RCDASW=BIT3           ;READ CHECK DATA SWITCH
2909          000020              OREQSW=BIT4           ;OFFSET REQUIRED SWITCH
2910                                      .EVEN
2911
2912          000114              MEMVEC=114            ;MEMORY PARITY TRAP VECTOR
2913  003150  000000              SAVWRD: .WORD   0     ;SCRATCH WORD
2914  003152  000400              BSSOFT: .BLKW   ↑D256 ;RECORD OF BAD SECTORS FROM SOFTWARE
2915  004152  000400              BSFACT: .BLKW   ↑D256 ;RECORD OF BAD SECTORS FROM FACTORY
2916  005152  000000  000000  000000  PRVCMD: .WORD   0,0,0,0,0,0  ;PREVIOUS COMMAND STORAGE
2917  005160  000000  000000  000000
2918  005166  000000  000000  000000  COMSTR: .WORD   0,0,0,0,0,0  ;CURRENT COMMAND STORAGE
2919  005174  000000  000000  000000
2920  005202  000102              BUFFO:  .BLKW   ↑D66  ;OUTPUT BUFFER 1
2921  005406  000000              LOWOCT: .WORD   0     ;LOW 16 BITS OF CONVERTED BINARY NUMBER
2922  005410  000000              HIGOCT: .WORD   0     ;HIGH BITS OF CONVERTED BINARY NO.
2923  005412  000000              BUFPRT: .WORD   0     ;BUFFER POINTER
2924  005414  000000              RECODE: .WORD   0     ;RECOVERY CODE WORD
2925  005416  000000              ERRCOM: .WORD   0     ;ERROR COMMAND
2926  005420  000000              DRIVE:  .WORD   0     ;NO.  OF  DRIVE IN USE
2927  005422  000000              STALLS: .WORD   0     ;CURRENT NO. OF UNIT STALLS TO APPLY
2928  005424  000000              CYLNDR: .WORD   0     ;CURRENT CYLINDER NUMBER
2929  005426  000000              OFINUS: .WORD   0     ;OFFSET IN USE
2930  005430  000000              TRACK:  .WORD   0     ;TRACK IN USE
2931  005432  000000              INTCHR: .WORD   0     ;TTY INTERRUPT INPUT WORD
2932  005434  000000              SCRACH: .WORD   0     ;ALL-PURPOSE SCRATCH WORD
2933  005436  000000              PATRN:  .WORD   0     ;DATA PATTERN LIST WORD
2934  005440  000000              PLOFST: .WORD   0     ;(+) OFFSET VALUE
2935  005442  000000              NGOFST: .WORD   0     ;(-) OFFSET VALUE
2936  005444  000005              SAVPRS: .BLKW   5     ;SAVE INITIAL PARAMS FOR XFER
2937  005456  000000              WDSXFR: .WORD   0     ;NO. OF WORDS ACTUALLY XFERRED
2938  005460  000000              FINCYL: .WORD   0     ;FINAL CYLINDER ADRS
2939  005462      000             FINTRK: .BYTE   0     ;FINAL TRACK ADRS
2940  005463      000             FINSEC: .BYTE   0     ;FINAL SECTOR ADRS
2941  005464  000000              LASTWC: .WORD   0     ;ACTUAL FINAL WC
2942  005466  000000              NEWON:  .WORD   0     ;NEW LOOK AT ONLINE DRIVES
2943  005470  000002              MA:     .BLKW   2
2944  005474  000002              PMA:    .BLKW   2     ;MEM. BUFFER ADDRESS
2945  005500  000000              SUBSYS: .WORD   0     ;CURRENT SUBSYSTEM NAME STORAGE (ASCII)
2946  005502  000021              DRVLST: .BLKW   17.   ;EACH WORD CONTAINS THE NAME OF A DRIVE
2947                                                    ;  TO BE TESTED. HIGH BYTE IS SUBSYSTEM
2948                                                    ;  NAME, LOW BYTE IS DRIVE NO. (ASCII).
2949                                                    ;  IF WORD = 0, DRIVE IS NOT TESTED.
2950  005544  000000              DRVPTR: .WORD   0     ;DRIVE LIST POINTER
2951  005546  000010              SCRLST: .BLKB   8.    ;SCRATCH DRIVE LIST
2952
2953  005556                      PAT00:
```

```
2954
2955
2956
2957
2958
2959  005556
2960  005556  000620  000000          ;TABLE A  -  BASIC READ/WRITE TEST SECTORS
2961  005562  000620  000002          TABLEA:
2962  005564  000620  000004                .WORD   620,0           ;DRIVE  0
2963  005572  000620  000006                .WORD   620,2           ;       1
2964  005576  000620  000010                .WORD   620,4           ;       2
2965  005602  000620  000012                .WORD   620,6           ;       3
2966  005606  000620  000014                .WORD   620,10          ;       4
2967  005612  000620  000016                .WORD   620,12          ;       5
2968  005616  000622  000000                .WORD   620,14          ;       6
2969  005622  000622  000002                .WORD   620,16          ;       7
2970  005626  000622  000004                .WORD   622,0           ;       10
2971  005632  000622  000006                .WORD   622,2           ;       11
2972  005636  000622  000010                .WORD   622,4           ;       12
2973  005642  000622  000012                .WORD   622,6           ;       13
2974  005646  000622  000014                .WORD   622,10          ;       14
2975  005652  000622  000016                .WORD   622,12          ;       15
2976                                        .WORD   622,14          ;       16
2977                                        .WORD   622,16          ;       17
2978
2979                                ;TABLE B  -  WORST CASE DATA PATTERN
2980  005656                        TABLEB:
2981  005656  072307                        072307
2982  005660  135143                        135143
2983  005662  156461                        156461
2984  005664  167230                        167230
2985  005666  073514                        073514
2986  005670  035646                        035646
2987  005672  016723                        016723
2988  005674  107351                        107351
2989  005676  143564                        143564
2990  005700  061672                        061672
2991  005702  030735                        030735
2992  005704  114356                        114356
2993  005706  046167                        046167
2994  005710  123073                        123073
2995  005712  151453                        151453
2996  005714  164616                        164616
2997
2998
2999
3000                                ;TABLE OF STARTING CYLINDERS FOR ALL OVWRT AND COMPAT CYL BLKS
3001  005716  000000  000060          TABLEC: .WORD   0,60            ;FOR CURRENT ZONE 1
3002  005722  000100  000160                .WORD   100,160         ;                 2
3003  005726  000200  000260                .WORD   200,260         ;                 3
3004  005732  000300  000360                .WORD   300,360         ;                 4
3005  005736  000400  000460                .WORD   400,460         ;                 5
3006  005742  000500  000560                .WORD   500,560         ;                 6
3007  005746  000600                        .WORD   600             ;                 7
3008
3009
```

```
3010
3011
3012  005750  000020                ;CYLINDER BLOCK LAYOUT TABLE
3013  005770  000020  TABLED: .BLKB   16.          ;CYLINDER 0 IN BLK
3014  006010  000020         .BLKB   16.          ;         1
3015  006030  000020         .BLKB   16.          ;         2
3016  006050  000020         .BLKB   16.          ;         3
3017  006070  000020         .BLKB   16.          ;         4
3018  006110  000020         .BLKB   16.          ;         5
3019  006130  000020         .BLKB   16.          ;         6
3020  006150  000020         .BLKB   16.          ;         7
3021  006170  000020         .BLKB   16.          ;        10
3022  006210  000020         .BLKB   16.          ;        11
3023  006230  000020         .BLKB   16.          ;        12
3024  006250  000020         .BLKB   16.          ;        13
3025  006270  000020         .BLKB   16.          ;        14
3026  006310  000020         .BLKB   16.          ;        15
3027  006330  000020         .BLKB   16.          ;        16
3028                                             ;        17
3029
3030
3031                          ;TABLE G - PSEUDO-RANDOM DATA PATTERN
3032  006350                  TABLEG:
3033  006350  040135                  040135
3034  006352  177070                  177070
3035  006354  070414                  070414
3036  006356  064531                  064531
3037  006360  174473                  174473
3038  006362  062422                  062422
3039  006364  114352                  114352
3040  006366  036620                  036620
3041  006370  010031                  010031
3042  006372  052336                  052336
3043  006374  017310                  017310
3044  006376  011347                  011347
3045  006400  102367                  102367
3046  006402  152567                  152567
3047  006404  001246                  001246
3048  006406  160073                  160073
3049
3050
3051
3052                          ;LIST OF STARTING DRIVE NUMBERS, TO GENERATE TABLED
3053  006410  000  001  003   STDLST: .BYTE   0,1,3,6,12,17,5,14,4,15,7,2,16,13,11,10
3054  006413  006  012  017
3055  006416  005  014  004
3056  006421  015  007  002
3057  006424  016  013  011
3058  006427  010
3059
3060
3061
3062                          ;LIST OF USEABLE SECTORS IN EACH CYLINDER BLOCK
3063  006430  000  001  002   SECLST: .BYTE   0,1,2,3,5,6,7,10,12,13,14,15,17,20,21,22
3064  006433  003  005  006
3065  006436  007  010  012
```

```
3066  006441    013     014     015
3067  006444    017     020     021
3068  006447    022
3069
3070
3071
3072                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR NEG OFST
3073  006450  000020                NOSCR0: .BLKW   16.
3074
3075                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR NEG OFST
3076  006510  000020                NOSCR1: .BLKW   16.
3077
3078                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR NEG OFST
3079  006550  000020                NOSCR2: .BLKW   16.
3080
3081
3082
3083                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR POS OFST
3084  006610  000020                POSCR0: .BLKW   16.
3085
3086                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR POS OFST
3087  006650  000020                POSCR1: .BLKW   16.
3088
3089                                ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR POS OFST
3090  006710  000020                POSCR2: .BLKW   16.
3091
3092
3093
3094                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR NEG OFST
3095  006750  000020                NCSCR0: .BLKW   16.
3096
3097                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR NEG OFST
3098  007010  000020                NCSCR1: .BLKW   16.
3099
3100                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR NEG OFST
3101  007050  000020                NCSCR2: .BLKW   16.
3102
3103
3104
3105                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR POS OFST
3106  007110  000020                PCSCR0: .BLKW   16.
3107
3108                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR POS OFST
3109  007150  000020                PCSCR1: .BLKW   16.
3110
3111                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR POS OFST
3112  007210  000020                PCSCR2: .BLKW   16.
3113
3114
3115
3116
3117                                ;SELF-TEST SCORE FOR TRACK 0 FOR NEG OFST
3118  007250  000000                NSSCR0: .WORD   0
3119
3120                                ;SELF-TEST SCORE FOR TRACK 1 FOR NEG OFST
3121  007252  000000                NSSCR1: .WORD   0
```

```
                                        ;SELF-TEST SCORE FOR TRACK 2 FOR NEG OFST
007254  000000                          NSSCR2: .WORD   0

                                        ;SELF-TEST SCORE FOR TRACK 0 FOR POS OFST
007256  000000                          PSSCR0: .WORD   0

                                        ;SELF-TEST SCORE FOR TRACK 1 FOR POS OFST
007260  000000                          PSSCR1: .WORD   0

                                        ;SELF-TEST SCORE FOR TRACK 2 FOR POS OFST
007262  000000                          PSSCR2: .WORD   0


                                        ;AVERAGE OF OVRWRT SCORES OVER ALL SURFACES
007264  000000                          OVRAVG: .WORD   0

                                        ;AVERAGE OF READ SCORES OVER ALL SURFACES
007266  000000                          COMAVG: .WORD   0


                                        ;SCRATCH TABLE OF SCORES FOR TRACK 0 FOR NEG OFST
007270  000020                          NSCOR0: .BLKB   16.

                                        ;SCRATCH TABLE OF SCORES FOR TRACK 1 FOR NEG OFST
007310  000020                          NSCOR1: .BLKB   16.

                                        ;SCRATCH TABLE OF SCORES FOR TRACK 2 FOR NEG OFST
007330  000020                          NSCOR2: .BLKB   16.


                                        ;SCRATCH TABLE OF SCORES FOR TRACK 0 FOR POS OFST
007350  000020                          PSCOR0: .BLKB   16.

                                        ;SCRATCH TABLE OF SCORES FOR TRACK 1 FOR POS OFST
007370  000020                          PSCOR1: .BLKB   16.

                                        ;SCRATCH TABLE OF SCORES FOR TRACK 2 FOR POS OFST
007410  000020                          PSCOR2: .BLKB   16.


        000632                          LSTCYL=632                      ;LAST CYL. = 410(10)
        000002                          LSTTRK=2                        ;LAST TRACK  = 2
        000002                          BSERR=BIT1                      ;BSE ERROR
        000004                          HVRCER=BIT2                     ;HVRC ERROR
        000010                          OPIERR=BIT3                     ;OPI ERROR
        000020                          DCKERR=BIT4                     ;DATA CHECK ERROR
        000040                          ECCNC=BIT5                      ;ECC NON-CORRECTABLE
        000100                          WCERR=BIT6                      ;WRITE CHECK ERROR
        000200                          ABORT=BIT7                      ;ABORT
```

```
3178                 000400                 LEV2ER=BIT8                    ;LEVEL TWO ERROR
3179                 001000                 BADSEC=BIT9                    ;BAD SECTOR FLAG
3180                 002000                 TWOTOS=BIT10                   ;TWO TIME OUTS
3181                 004000                 RCLREQ=BIT11                   ;RECALIBRATE REQUIRED
3182                 010000                 DRNAVL=BIT12                   ;DRIVE NOT RELSD BY OTHER PORT
3183                 100000                 ANYDER=BIT15                   ;ANY ERROR DETECTED FLAG
3184
3185  007430  005015  046412  026504  DZR6Q:  .ASCIZ  <15><12><12>/MD-11-DZR6Q-A - RK06 DRIVE COMPATIBILITY PROGRAM/<15><12><1
3186  007436  030461  042055  051132
3187  007444  050466  040455  026440
3188  007452  051040  030113  020066
3189  007460  051104  053111  020105
3190  007466  047503  050115  052101
3191  007474  041111  046111  052111
3192  007502  020131  051120  043517
3193  007510  040522  006515  005012
3194  007516     000
3195  007517     012  045522  033060  RKBADR: .ASCIZ  <12>/RK06 BUS ADR = /
3196  007524  041040  051525  040440
3197  007532  051104  036440  000040
3198  007540  045522  033060  053040  RKVADR: .ASCIZ  /RK06 VEC ADR = /
3199  007546  041505  040440  051104
3200  007554  036440  000040
3201  007560  045522  033060  050040  RKPRTY: .ASCIZ  /RK06 PRIORITY =/
3202  007566  044522  051117  052111
3203  007574  020131  000075
3204  007600  052012  050131  020105  TYPSYS: .ASCIZ  <12>/TYPE NAME OF NEXT SUBSYS TO TEST (A-P) : /
3205  007606  040516  042515  047440
3206  007614  020106  042516  052130
3207  007622  051440  041125  054523
3208  007630  020123  047524  052040
3209  007636  051505  020124  040450
3210  007644  050055  020051  020072
3211  007652     000
3212  007653     123  041125  054523  SYSDRV: .ASCIZ  /SUBSYS   DRIVE(S) = /
3213  007660  020123  020040  051104
3214  007666  053111  024105  024523
3215  007674  036440  000040
3216  007700  044527  046114  052040  WILTST: .ASCIZ  /WILL TEST DRIVE(S) /
3217  007706  051505  020124  051104
3218  007714  053111  024105  024523
3219  007722  000040
3220  007724  047440  020116  052523  ONSUBS: .ASCIZ  / ON SUBSYS  /<15><12><12>
3221  007732  051502  051531  020040
3222  007740  005015  000012
3223  007744  051511  052040  042510  ANOTHR: .ASCIZ  /IS THERE ANOTHER SUBSYS (Y OR N <CR>) ? /
3224  007752  042522  040440  047516
3225  007760  044124  051105  051440
3226  007766  041125  054523  020123
3227  007774  054450  047440  020122
3228  010002  020116  041474  037122
3229  010010  020051  020077     000
3230  010015     012  047515  042522  DROVFL: .ASCIZ  <12>/MORE THAN 16 DRIVES ENTERED !/<15><12><12>
3231  010022  052040  040510  020116
3232  010030  033061  042040  044522
3233  010036  042526  020123  047105
```

```
 3234  010044  042524  042522  020104
 3235  010052  006441  005012     000
 3236  010057     012  025052  051440  SRPASS: .ASCIZ  <12>/** STARTING PASS   **/<15><12>
 3237  010064  040524  052122  047111
 3238  010072  020107  040520  051523
 3239  010100  020040  025040  006452
 3240  010106  000012
 3241  010110  046412  052517  052116  MNTPAK: .ASCIZ  <12>/MOUNT PACK ON DRIVE    AND LOAD./<15><12>
 3242  010116  050040  041501  020113
 3243  010124  047117  042040  044522
 3244  010132  042526  020040  020040
 3245  010140  047101  020104  047514
 3246  010146  042101  006456  000012
 3247  010154  052412  046116  040517  DISPAK: .ASCIZ  <12>/UNLOAD DRIVE    AND REMOVE PACK./<15><12>
 3248  010162  020104  051104  053111
 3249  010170  020105  020040  040440
 3250  010176  042116  051040  046505
 3251  010204  053117  020105  040520
 3252  010212  045503  006456  000012
 3253  010220  051412  040524  052122  STR204: .ASCIZ  <12>/START AT ADR 204 FOR PASS 1 ON SUBSYS  ./<15><12><12>
 3254  010226  040440  020124  042101
 3255  010234  020122  030062  020064
 3256  010242  047506  020122  040520
 3257  010250  051523  030440  047440
 3258  010256  020116  052523  051502
 3259  010264  051531  020040  006456
 3260  010272  005012     000
 3261  010275     012  052123  051101  STR220: .ASCIZ  <12>/START AT ADR 220 FOR PASS 2 ON SUBSYS  ./<15><12><12>
 3262  010302  020124  052101  040440
 3263  010310  051104  031040  030062
 3264  010316  043040  051117  050040
 3265  010324  051501  020123  020062
 3266  010332  047117  051440  041125
 3267  010340  054523  020123  027040
 3268  010346  005015  000012
 3269  010352  054524  042520  051040  TYPRWN: .ASCIZ  /TYPE R<CR> WHEN /
 3270  010360  041474  037122  053440
 3271  010366  042510  020116     000
 3272  010373     104  044522  042526  DRIRDY: .ASCIZ  /DRIVE READY : /
 3273  010400  051040  040505  054504
 3274  010406  035040  000040
 3275  010412  047504  042516  035040  DRIDON: .ASCIZ  /DONE : /
 3276  010420  000040
 3277  010422  047516  042040  044522  NODRLS: .ASCIZ  /NO DRIVES LISTED FOR THIS SUBSYS !/<15><12>
 3278  010430  042526  020123  044514
 3279  010436  052123  042105  043040
 3280  010444  051117  052040  044510
 3281  010452  020123  052523  051502
 3282  010460  051531  020440  005015
 3283  010466     000
 3284  010467     015  051412  047503  RESLTS: .ASCII  <15><12>/SCORES FOR DRIVE     (0-12 DEC.) :/<15><12><12>
 3285  010474  042522  020123  047506
 3286  010502  020122  051104  053111
 3287  010510  020105  020040  020040
 3288  010516  030050  030455  020062
 3289  010524  042504  027103  020051
```

```
3290  010532  006472  005012
3291  010536  051124  041501  004513          .ASCII  /TRACK  DRIVE   OVRWRT  OVRWRT  READ    READ/<15><12>
3292  010544  051104  053111  004505
3293  010552  053117  053522  052122
3294  010560  047411  051126  051127
3295  010566  004524  042522  042101
3296  010574  051011  040505  006504
3297  010602      012
3298  010603      116 046525  051011          .ASCII  /NUM    READ    SCORE   SCORE   SCORE   SCORE/<15><12>
3299  010610  040505  004504  041523
3300  010616  051117  004505  041523
3301  010624  051117  004505  041523
3302  010632  051117  004505  041523
3303  010640  051117  006505      012
3304  010645      011 047411  051506          .ASCIZ  /               OFST-   OFST+   OFST-   OFST+/<15><12><12>
3305  010652  026524  047411  051506
3306  010660  025524  047411  051506
3307  010666  026524  047411  051506
3308  010674  025524  005015  000012
3309  010702  047440  051126  051127  OAVERG: .ASCIZ  / OVRWRT AVG = /
3310  010710  020124  053101  020107
3311  010716  020075      000
3312  010721      040 042522  042101  RAVERG: .ASCIZ  / READ AVG   = /
3313  010726  040440  043526  020040
3314  010734  036440  000040
3315  010740  005015  025012  020052  ENDTST: .ASCIZ  <15><12><12>/**  END OF TESTING  **/<15><12>
3316  010746  042440  042116  047440
3317  010754  020106  042524  052123
3318  010762  047111  020107  025040
3319  010770  006452  000012
3320  010774  020040  000011          TRKBUF: .ASCIZ  /       /
3321  011000  020040  004440      000 DRVBUF: .ASCIZ  /       /
3322  011005      123 046105  004506  SELF:   .ASCIZ  /SELF   /
3323  011012      000
3324  011013      011     000         TAB:    .ASCIZ  /       /
3325  011015      123 051127  036440  SWRMSG: .ASCIZ  /SWR = /
3326  011022  000040
3327  011024  020040  042516  020127  NEWMSG: .ASCIZ  /  NEW = /
3328  011032  020075      000
3329  011035      015 042012  044522  DRVSEQ: .ASCIZ  <15><12>/DRIVE(S) = /
3330  011042  042526  051450  020051
3331  011050  020075      000
3332  011053      104 044522  042526  BADDRV: .ASCIZ  /DRIVE   /
3333  011060  020040  000040
3334  011064  047516  026516  054105  NXDRIV: .ASCIZ  /NON-EXISTENT/<15><12>
3335  011072  051511  042524  052116
3336  011100  005015      000
3337  011103      116 052117  051040  NTREDY: .ASCIZ  /NOT READY/<15><12>
3338  011110  040505  054504  005015
3339  011116      000
3340  011117      127 044522  042524  WRTLOK: .ASCIZ  /WRITE-LOCKED/<15><12>
3341  011124  046055  041517  042513
3342  011132  006504  000012
3343  011136  047514  042101  042105  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
3344  011144  053440  052111  020110
3345  011152  046101  043511  020116
```

```
3346  011160  040520  045503  005015
3347  011166     000
3348  011167     111  020106  054130  REPLPK: .ASCIZ  /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", &  REPLACE IT : /
3349  011174  050104  050040  041501
3350  011202  020113  047117  042040
3351  011210  053122  030040  020054
3352  011216  054524  042520  021040
3353  011224  020131  041474  037122
3354  011232  026042  023040  020040
3355  011240  042522  046120  041501
3356  011246  020105  052111  035040
3357  011254  000040
3358  011256  005015  025012  020052  NODRTS: .ASCII  <15><12><12>/** NO DRIVES TO TEST/<15><12>
3359  011264  047516  042040  044522
3360  011272  042526  020123  047524
3361  011300  052040  051505  006524
3362  011306     012
3363  011307     120  042522  051523  CNTRDY: .ASCIZ  /PRESS "CONT" WHEN RDY/<15><12>
3364  011314  021040  047503  052116
3365  011322  020042  044127  047105
3366  011330  051040  054504  005015
3367  011336     000
3368  011337     110  046101  020124  HLTRQD: .ASCIZ  /HALT REQUESTED/<15><12>
3369  011344  042522  052521  051505
3370  011352  042524  006504  000012
3371  011360  051104  053111  020105  DROXDP: .ASCIZ  /DRIVE 0 IS LOAD MEDIUM/<15><12>
3372  011366  020060  051511  046040
3373  011374  040517  020104  042515
3374  011402  044504  046525  005015
3375  011410     000
3376  011411     015  025012  020052  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRIVE - 20 ERRORS/<15><12>
3377  011416  042040  047522  050120
3378  011424  047111  020107  051104
3379  011432  053111  020105  020055
3380  011440  030062  042440  051122
3381  011446  051117  006523  000012
3382  011454  005015  052012  051505  TSTDRN: .ASCII  <15><12><12>/TESTING DRIVE /
3383  011462  044524  043516  042040
3384  011470  044522  042526     040
3385  011475     040  005015     000  DRVNO:  .ASCIZ  / /<15><12>
3386  011501     104  044522  042526  DRIV:   .ASCIZ  /DRIVE /
3387  011506  000040
3388  011510  040503  052122  020056  CART:   .ASCIZ  /CART. /
3389  011516     000
3390  011517     123  051105  020056  SERNM:  .ASCIZ  /SER. NO.  /
3391  011524  047516  020056  000040
3392  011532  005015  040506  052103  FACTBS: .ASCIZ  <15><12>/FACTORY /
3393  011540  051117  020131     000
3394  011545     012  047523  052106  SOFTBS: .ASCII  <12>/SOFTWARE /
3395  011552  040527  042522     040
3396  011557     102  042101  051440  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
3397  011564  041505  047524  051522
3398  011572  035040  005015     000
3399  011577     040  047040  047117  NOFALS: .ASCIZ  /  NONE/
3400  011604  000105
3401
```

```
3402  011606  025052  020040  040503  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRACK!/<15><12>
3403  011614  047116  052117  051040
3404  011622  040505  020104  040502
3405  011630  020104  042523  052103
3406  011636  051117  052040  040522
3407  011644  045503  006441  000012
3408  011652  041536  005015     000  CNTRLC: .ASCIZ  /↑C/<15><12>
3409  011657     136  006532  000012  CNTRLZ: .ASCIZ  /↑Z/<15><12>
3410  011664  051136  005015     000  CNTRLR: .ASCIZ  /↑R/<15><12>
3411  011671     136  006525  000012  CNTRLU: .ASCIZ  /↑U/<15><12>
3412  011676  043536  005015     000  CNTRLG: .ASCIZ  /↑G/<15><12>
3413  011703     015  005012     000  CR2LF:  .ASCIZ  <15><12><12>
3414  011707     134     000          BKSLSH: .ASCIZ  /\/
3415  011711     054     000          COMMA:  .ASCIZ  /,/
3416  011713     040     040          SPACE6: .ASCII  /  /
3417  011715     040                  SPACE4: .ASCII  / /
3418  011716     040                  SPACE3: .ASCII  / /
3419  011717     040                  SPACE2: .ASCII  / /
3420  011720  000040                  SPACE1: .ASCIZ  / /
3421                                          .EVEN
3422  011722  020040  000075          PRMBUF: .ASCIZ  /  =/
3423  011726  047527  042122  000040  WORDSP: .ASCIZ  /WORD /
3424  011734  036440  000040  EQUALS: .ASCIZ  / = /
3425  011740  020052     000  PROMPT: .ASCIZ  /* /
3426  011743     076  000040  PRMPSP: .ASCIZ  /> /
3427
3428                                          .EVEN
3429
3430
3431
3432
3433  011746  105037  003116  DFSTRT: CLRB    MDFLAG          ;SET FLAG FOR 200 START
3434  011752  000403          P1SET:  BR      P1SET           ;CONTINUE
3435
3436  011754  112737  000001  003116  P1STRT: MOVB    #1,MDFLAG       ;SET FLAG FOR 204 START
3437  011762  112737  000061  003145  P1SET:  MOVB    #'1,PASSNO      ;SET PASS NO. = 1
3438  011770  000406          P2STRT: BR      CMSTRT          ;CONTINUE
3439
3440  011772  112737  000002  003116          MOVB    #2,MDFLAG       ;SET FLAG FOR 220 START
3441  012000  112737  000062  003145          MOVB    #'2,PASSNO      ;SET PASS NO. = 2
3442
3443  012006  012737  000340  177776  CMSTRT: MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
3444                                  .SBTTL  INITIALIZE THE COMMON TAGS
3445                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3446  012014  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3447  012020  005026                  1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
3448  012022  022706  001140                  CMP     #SWR,R6 ;;DONE?
3449  012026  001374                          BNE     1$              ;;LOOP BACK IF NO
3450  012030  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
3451                                  ;;INITIALIZE A FEW VECTORS
3452  012034  012737  043742  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3453  012042  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
3454  012050  012737  043236  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3455  012056  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
3456  012064  012737  044454  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3457  012072  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
```

```
3458  012100  012737  043606  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3459  012106  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;LEVEL 7
3460  012114  013737  016676  016670          MOV     $ENDCT,$EOPCT    ;;SETUP END-OF-PROGRAM COUNTER
3461  012122  005037  001316                  CLR     $ESCAPE          ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3462  012126  112737  000001  001115          MOVB    #1,$ERMAX        ;;ALLOW ONE ERROR PER TEST
3463  012134  012737  012134  001106          MOV     #.,$LPADR        ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3464  012142  012737  012142  001110          MOV     #.,$LPERR        ;;SETUP THE ERROR LOOP ADDRESS
3465                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3466                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
3467  012150  013746  000004                  MOV     @#ERRVEC,-(SP)   ;;SAVE ERROR VECTOR
3468  012154  012737  012210  000004          MOV     #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
3469  012162  012737  177570  001140          MOV     #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
3470  012170  012737  177570  001142          MOV     #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
3471  012176  022777  177777  166734          CMP     #-1,@SWR         ;;TRY TO REFERENCE HARDWARE SWR
3472  012204  001012                          BNE     66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3473                                                                   ;;AND  THE HARDWARE SWR IS NOT = -1
3474  012206  000403                          BR      65$              ;;BRANCH IF NO TIMEOUT
3475  012210  012716  012216          64$:    MOV     #65$,(SP)        ;;SET UP FOR TRAP RETURN
3476  012214  000002                          RTI
3477  012216  012737  000176  001140  65$:    MOV     #SWREG,SWR       ;;POINT TO SOFTWARE SWR
3478  012224  012737  000174  001142          MOV     #DISPREG,DISPLAY
3479  012232  012637  000004          66$:    MOV     (SP)+,@#ERRVEC   ;;RESTORE ERROR VECTOR
3480
3481  012236  005037  001336                  CLR     $PASS            ;;CLEAR PASS COUNT
3482  012242  132737  000200  001351          BITB    #APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
3483  012250  001403                          BEQ     67$              ;;YES,USE NON-APT SWITCH
3484  012252  012737  001352  001140          MOV     #$SWREG,SWR      ;;NO,USE APT SWITCH REGISTER
3485  012260                          67$:
3486  012260  000005                          RESET                    ;CLEAR THE UNIBUS
3487  012262  012737  000006  000004          MOV     #6,@#ERRVEC      ;SET TIME-OUT VECTOR
3488  012270  005037  000006                  CLR     @#ERRVEC+2
3489  012274  105037  003133                  CLRB    TSTTYP           ;CLEAR OFFSET TEST IDENTIFIER
3490  012300  105037  003127                  CLRB    DRVERS           ;CLEAR ERROR COUNT FOR CURRENT DRIVE
3491  012304  105037  003125                  CLRB    ERRCNT           ;CLEAR ERROR RETRY COUNT
3492  012310  105037  001102                  CLRB    STSTNM           ;SET TEST NO. = 0
3493  012314  112737  000001  003142          MOVB    #1,HLPOVL        ;SET HLP FILE OVLD INDICTR
3494  012322  104401  007430                  TYPE    ,DZR6Q           ;TYPE PROGRAM I.D.
3495  012326  105737  003143                  TSTB    NOTYPE           ;SEE IF OPERATOR NOTE SHOULD BE TYPED
3496  012332  001004                          BNE     40$              ;BR IF NOT
3497  012334  104401  053476                  TYPE    ,NOTMSG          ;TYPE OPERATOR NOTE
3498  012340  105237  003143                  INCB    NOTYPE           ;SET FLAG FOR NEXT TIME
3499  012344  012737  020262  000060  40$:    MOV     #KBDHDL,@#TKVEC  ;LOAD VECTOR FOR TTY KBD
3500  012352  012737  000200  000062          MOV     #PR4,@#TKVEC+2   ;SET KBD PRIORITY = 4
3501  012360  013701  003040                  MOV     RKVEC,R1         ;ADDR. OF RK06 VECTOR STORAGE
3502  012364  012721  034056                  MOV     #I.INTR,(R1)+    ;SET IT TO RK06 HANDLER
3503  012370  013711  003042                  MOV     RKPRI,(R1)       ;SET RK06 PRIORITY
3504  012374  105037  003117                  CLRB    TSTING           ;CLEAR "RUNNING TESTS" FLAG
3505  012400  012737  000000  177776          MOV     #PR0,@#PS        ;ALLOW ALL INTERRUPTS AGAIN
3506  012406  012701  005152                  MOV     #PRVCMD,R1       ;ZERO OUT PREVIOUS COMMAND
3507  012412  012700  000006                  MOV     #6,R0            ;SET COUNTER
3508  012416  005021                  42$:    CLR     (R1)+            ;ZERO A WORD
3509  012420  005300                          DEC     R0
3510  012422  001375                          BNE     42$              ;BR IF NOT DONE YET
3511  012424  005037  005422                  CLR     STALLS           ;DON'T ALLOW STALLS YET
3512  012430  004737  020564                  JSR     PC,GTSWRG        ;OPEN SOFTWARE SWR FOR MODIFICATION
3513  012434  012737  176543  041504  44$:    MOV     #176543,$HINUM   ;INIT. PSEUDO-RANDOM NOS.
```

```
                                        MOV     #123456,$LONUM
                                9$:     TSTB    MDFLAG          ;SEE IF DEFAULT MODE
                                        BNE     ALLSYS          ;BR IF NOT DEFAULT MODE
                                        MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
                                        MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
                                        BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
                                        BNE     18$             ;BR IF NO SIZING
                                        MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
                                        MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
                                18$:    MOVB    R0,RKVEC        ;STORE VECTOR
                                        CLRB    RKVEC+1         ;CLEAR HI BYTE
                                        SWAB    R0              ;GET PRTY INTO BITS 5-7
                                        BIC     #177437,R0      ;CLEAR OTHER BITS
                                        MOV     R0,RKPRI        ;STORE RK06 PRIORITY
                                        MOVB    #'A,SUBSYS      ;SET SUBSYSTEM NAME = A
                                        JMP     DFLTDR          ;GO CHECK ALL DRIVES


;;*****************************************************************
;*
.SBTTL  TEST 0     PARAMETER INPUT ROUTINES
;*
;;*****************************************************************
;READ AND STORE ALL SUBSYSTEMS AND DRIVES TO TEST
ALLSYS: TYPE    ,$CRLF          ;TYPE <CR>,<LF>
        CLRB    $TSTNM          ;SET TEST NO. = 0
        MOVB    #'A,R3          ;INIT SUBSYS INDICATOR
        MOV     #DRVLST,R1      ;GET DRIVE LIST ADDRESS
        MOV     #17.,R0         ;CLEAR OUT THE DRIVE LIST
2$:     CLR     (R1)+
        DEC     R0
        BNE     2$
        MOV     #DRVLST,R4      ;GET STARTING ADRS OF DRIVE LIST
;ASK FOR DRIVES TO TEST ON THIS SUBSYSTEM
ALLSY1: MOVB    R3,SYSDRV+7     ;SET SUBSYS NAME FOR PRINTOUT
6$:     TYPE    ,SYSDRV         ;ASK FOR DRIVES ON THIS SUBSYSTEM
        JSR     PC,RDCHRS       ;READ INPUT STRING
        ALLSYS                  ;(^C) RETURN ADDRESS
        ALLSYS                  ;(^Z) RETURN ADDRESS
        6$                      ;(^U) RETURN ADDRESS
        TST     R0              ;SEE IF NULL INPUT
        BEQ     6$              ;BR TO ASK AGAIN
        CMP     #15.,R0         ;SEE IF TOO MANY CHARS TYPED
        BGE     12$             ;BR IF NOT TOO MANY
10$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
        TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
        BR      6$              ;GO ASK AGAIN
12$:    CLR     R1              ;CLEAR CHAR POINTER
14$:    CMPB    BUFFO(R1),#'0   ;SEE IF DRIVE NO. < 0
        BLO     10$             ;BR TO ECHO BAD INPUT
        CMPB    BUFFO(R1),#'7   ;SEE IF CHAR > 7
        BHI     10$             ;BR TO ECHO BAD INPUT
        INC     R1              ;INCR CHAR POINTER
        CMP     R1,R0           ;SEE IF MORE CHARS TO CHECK
        BEQ     16$             ;BR IF ALL CHARS CHECKED
```

```
3570  012702  126127  005202  000054          CMPB    BUFFO(R1),#',   ;SEE IF THIS IS A COMMA
3571  012710  001353                          BNE     10$             ;BR IF NOT A COMMA
3572  012712  005201                          INC     R1              ;INCR CHAR POINTER
3573  012714  000757                          BR      14$             ;BR TO CONTINUE CHECKING CHARS
3574                          ;PUT DRIVES FOR THIS SUBSYSTEM INTO SCRATCH LIST
3575  012716  012701  005546          16$:    MOV     #SCRLST,R1      ;GET SCRATCH DRIVE LIST ADRS
3576  012722  005021                          CLR     (R1)+           ;CLEAR OUT SCRATCH DRIVE LIST
3577  012724  005021                          CLR     (R1)+
3578  012726  005021                          CLR     (R1)+
3579  012730  005011                          CLR     (R1)
3580  012732  005000                          CLR     R0              ;INIT CHAR POINTER
3581  012734  116001  005202          18$:    MOVB    BUFFO(R0),R1    ;GET A DRIVE NO.
3582  012740  042701  000060                  BIC     #'0,R1          ;STRIP ASCII BITS
3583  012744  116061  005202  005546          MOVB    BUFFO(R0),SCRLST(R1)    ;SET ASCII DRV NO.
3584  012752  062700  000002                  ADD     #2,R0           ;POINT TO NEXT DRIVE NO.
3585  012756  105760  005201                  TSTB    BUFFO-1(R0)     ;SEE IF NULL CHAR YET
3586  012762  001364                          BNE     18$             ;BR IF NOT DONE YET
3587                          ;LOAD DRIVES FROM SCRATCH LIST INTO TOTAL DRIVE LIST
3588  012764  005000          ALLSY3: CLR     R0              ;INIT SCRATCH DRV LIST POINTER
3589  012766  116014  005546  ALLSY2: MOVB    SCRLST(R0),(R4) ;MOVE THIS DRIVE INTO ACTUAL LIST
3590  012772  001411                          BEQ     22$             ;BR IF DRIVE NOT MARKED
3591  012774  005204                          INC     R4              ;INCR DRIVE LIST POINTER
3592  012776  110324                          MOVB    R3,(R4)+        ;PUT SUBSYS LETTER INTO HI BYTE
3593  013000  020427  005544                  CMP     R4,#DRVLST+34.  ;SEE IF DRIVE LIST OVERFLOW
3594  013004  103404                          BLO     22$             ;BR IF NO OVERFLOW YET
3595  013006  104401  010015                  TYPE    ,DROVFL         ;TYPE "MORE THAN 16 DRIVES ENTERED !"
3596  013012  000137  012546                  JMP     ALLSYS          ;GO START OVER, ASKING FOR DRIVES
3597  013016  005200          22$:    INC     R0              ;INCR SCRATCH DRIVE LIST POINTER
3598  013020  020027  000010                  CMP     R0,#8.          ;SEE IF DONE WITH THIS SUBSYS
3599  013024  103760                          BLO     ALLSY2          ;BR IF NOT YET
3600                          ;ECHO DRIVES TO TEST ON THIS SUBSYSTEM
3601  013026  104401  007700                  TYPE    ,WILTST         ;TYPE "WILL TEST DRIVE(S) "
3602  013032  005037  005434                  CLR     $CRACH          ;INIT CHAR OUTPUT BUF
3603  013036  005000                          CLR     R0              ;INIT SCRATCH LIST POINTER
3604  013040  005001                          CLR     R1              ;INIT COUNT OF LISTED DRIVES FOR SUBSYS
3605  013042  116037  005546  005434  23$:    MOVB    SCRLST(R0),SCRACH ;SEE IF THIS DRIVE LISTED
3606  013050  001407                          BEQ     26$             ;BR IF DRIVE NOT LISTED
3607  013052  005701                          TST     R1              ;SEE IF FIRST TIME HERE
3608  013054  001402                          BEQ     24$             ;BR IF FIRST TIME
3609  013056  104401  011711                  TYPE    ,COMMA          ;TYPE A COMMA
3610  013062  104401  005434          24$:    TYPE    ,SCRACH         ;TYPE A DRIVE NO.
3611  013066  005201                          INC     R1              ;INCR. COUNT OF LISTED DRIVES FOR THIS SUBSYS
3612  013070  005200          26$:    INC     R0              ;INCR LIST POINTER
3613  013072  020027  000010                  CMP     R0,#8.          ;SEE IF DONE CHECKING LIST YET
3614  013076  002761                          BLT     23$             ;BR IF NOT DONE YET
3615  013100  110337  007737                  MOVB    R3,ONSUBS+11.   ;SET SUBSYS NO. IN MSG
3616  013104  104401  007724                  TYPE    ,ONSUBS         ;TYPE "ON SUBSYSTEM "
3617  013110  005203                          INC     R3              ;INCR SUBSYSTEM NAME
3618  013112  020327  000120                  CMP     R3,#'P          ;SEE IF NAME > P
3619  013116  101035                          BHI     ASKSYS          ;BR IF YES
3620  013120  105737  003116                  TSTB    MDFLAG          ;SEE IF 200 START
3621  013124  001002                          BNE     28$             ;BR IF  NOT
3622  013126  000137  013500                  JMP     DRVTST          ;START PASS 1
3623                          ;ASK IF THERE IS ANOTHER SUBSYSTEM TO SPECIFY
3624  013132  104401  007744          28$:    TYPE    ,ANOTHR         ;ASK FOR ANOTHER SUBSYS
3625  013136  004737  023006                  JSR     PC,RDCHRS       ;READ RESPONSE
```

```
                                                ALLSYS                  ;(↑C) RETURN ADDRESS
                                                ALLSYS                  ;(↑Z) RETURN ADDRESS
                                                28$                     ;(↑U) RETURN ADDRESS
                                                TST     R0              ;SEE IF NULL INPUT
                                                BEQ     ASKSYS          ;BR IF NULL INPUT, TO ASSUME <N>
                                                CMP     #'Y,BUFF0       ;SEE IF <Y> TYPED
                                                BNE     30$             ;BR IF NOT <Y>
                                                JMP     ALLSY1          ;GO ASK FOR DRIVES ON NEXT SUBSYS
                                        30$:    CMP     #'N,BUFF0       ;SEE IF <N> TYPED
                                                BEQ     ASKSYS          ;BR IF <N>
                                                TYPE    ,BUFF0          ;ECHO BAD INPUT
                                                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
                                                BR      28$             ;GO ASK AGAIN

                                        ;204 OR 220 START - GET NAME OF SUBSYSTEM TO TEST NEXT
                                        ASKSYS: CLRB    $TSTNM          ;SET TEST NO. = 0
                                                TYPE    ,TYPSYS         ;ASK FOR NEXT SUBSYS TO TEST
                                                JSR     PC,RDCHRS       ;READ RESPONSE
                                                ALLSYS                  ;(↑C) RETURN ADDRESS
                                                ASKSYS                  ;(↑Z) RETURN ADDRESS
                                                ASKSYS                  ;(↑U) RETURN ADDRESS
                                                TST     R0              ;SEE IF NULL INPUT
                                                BEQ     ASKSYS          ;BR IF NULL, TO ASK AGAIN
                                                CMP     BUFF0,#'A       ;SEE IF A CHAR < A TYPED
                                                BHIS    6$              ;BR IF NOT
                                        4$:     TYPE    ,BUFF0          ;ECHO BAD INPUT
                                                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
                                                BR      ASKSYS          ;GO ASK AGAIN
                                        6$:     CMP     BUFF0,#'P       ;SEE IF A CHAR > P TYPED
                                                BHI     4$              ;BR IF YES
                                                MOVB    BUFF0,SUBSYS    ;STORE CURRENT SUBSYSTEM NAME
                                                MOV     #DRVLST+1,R0    ;SET DRIVE LIST POINTER
                                        8$:     CMPB    (R0)+,SUBSYS    ;SEE IF A DRIVE LISTED FOR THIS SUBSYS
                                                BEQ     9$              ;BR IF YES
                                                INC     R0              ;INCR POINTER
                                                CMP     R0,#DRVLST+33.  ;SEE IF WHOLE LIST CHECKED YET
                                                BLO     8$              ;BR IF NOT YET
                                                TYPE    ,NODRLS         ;TYPE "NO DRIVES LISTED FOR SUBSYS"
                                                JMP     ASKSYS          ;GO  ASK FOR  A SUBSYS AGAIN
                                        9$:     SUB     #2,R0           ;POINT TO LOW BYTE OF LIST ENTRY
                                                MOV     R0,DRVPTR       ;STORE POINTER

                                        ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
                                                MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
                                                TYPE    ,RKBADR         ;TYPE "RK06 BUS ADR = "
                                                JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
                                                MOV     (SP)+,RKBAS     ;STORE NEW VALUE

                                        ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
                                                MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
                                                TYPE    ,RKVADR         ;TYPE "RK06 VEC ADR = "
                                                JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
                                                MOV     (SP)+,RKVEC     ;STORE NEW VALUE

                                        ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
                                        11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
                                                ASL     (SP)            ;GET IT INTO BITS 0-2
                                                ASL     (SP)
                                                ASL     (SP)
```

```
                                        SWAB    (SP)
                                        TYPE    ,RKPRTY         ;TYPE   "RK06 PRIORITY = "
                                        JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
                                        MOV     (SP)+,R0
                                        CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
                                        BLT     12$             ;BR IF NEW VALUE TOO SMALL
                                        CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
                                        BGT     12$             ;BR IF NEW VALUE TOO LARGE
                                        SWAB    R0              ;GET PRIORITY INTO BITS 5-7
                                        ASR     R0
                                        ASR     R0
                                        ASR     R0
                                        MOV     R0,RKPRI        ;STORE NEW VALUE
                                        TYPE    ,$CRLF
                                        BR      DRVTST
                                12$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
                                        TYPE    ,$QUES
                                        BR      11$             ;GO ASK AGAIN

                                DRVTST: MOVB    PASSNO,SRPASS+18.  ;SET PASS NO. FOR PRINTOUT
                                        TYPE    ,SRPASS         ;TYPE "STARTING PASS X"
                                        CMPB    #'1,PASSNO      ;SEE IF PASS 1
                                        BNE     2$              ;BR IF NOT PASS 1
                                        JMP     TST1            ;START PASS 1
                                2$:     MOVB    #4,$TSTNM       ;SET TEST NUMBER = 4
                                        JMP     TST5            ;START PASS 2


                                ;ADR 200 START - COMPILE SCRATCH LIST OF DRIVES TO SIZE
                                DFLTDR: CLR     R0              ;CLEAR DRIVE NUMBER
                                        MOV     $DEVM,R3        ;GET APT DEVICE MAP
                                        BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
                                        BNE     1$              ;BR IF NO SIZING
                                        MOV     #377,R3         ;SET UP FOR SIZING
                                1$:     MOV     #BIT0,R1        ;SET BIT POINTER
                                2$:     CLRB    SCRLST(R0)      ;INITIALIZE DRIVE ENTRY
                                        BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
                                        BEQ     4$              ;BR IF NOT
                                        MOVB    R0,SCRLST(R0)   ;MARK DRIVE IN SCRATCH LIST
                                        BISB    #'0,SCRLST(R0)  ;CONVERT NO. TO ASCII
                                4$:     ASL     R1              ;SHIFT BIT POINTER
                                        INC     R0              ;INCREMENT DRIVE NUMBER
                                        CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
                                        BNE     2$              ;BR IF NOT DONE YET
                                        BITB    #BIT7,$ENVM     ;SEE IF PROGRAM SHOULD SIZE
                                        BEQ     8$              ;BR IF YES
                                10$:    MOV     #DRVLST,R4      ;GET DRIVE LIST ADDRESS
                                        MOVB    SUBSYS,R3       ;GET SUBSYS NAME IN R3
                                        MOV     R4,R1           ;GET DRIVE LIST ADRS
                                        MOV     R4,DRVPTR       ;SET POINTER TO FIRST DRIVE TO TEST
                                        MOV     #17.,R0         ;SET COUNTER
                                11$:    CLR     (R1)+           ;CLEAR A DRIVE LIST WORD
                                        DEC     R0              ;DECREMENT COUNTER
                                        BNE     11$             ;BR IF NOT DONE YET
                                        JMP     ALLSY3          ;GO MOVE SCRATCH LIST INTO LIST
                                ;REMOVE NON-EXISTENT DRIVES FROM SCRATCH LIST
```

```
3738  013666  005000                  8$:   CLR     R0              ;CLEAR DRIVE NUMBER
3739  013670  105760  005546          12$:  TSTB    SCRLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN SCRATCH LIST
3740  013674  001467                        BEQ     14$             ;BR IF NOT MARKED
3741  013676  010037  005420                MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
3742  013702  004737  020616                JSR     PC,INITSS       ;CLEAR DRIVER AND INIT S.S.
3743  013706  042712  000100          3$:   BIC     #IE,(R2)        ;DISABLE RK06 INTERRUPT
3744  013712  113762  005420  000010        MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
3745  013720  012712  000001                MOV     #BIT0,(R2)      ;SELECT THE DRIVE
3746  013724  005037  005434                CLR     SCRACH          ;CLEAR STALL COUNTER
3747  013730  005237  005434          15$:  INC     SCRACH          ;INCREMENT STALL COUNTER
3748  013734  001375                        BNE     15$             ;STALL FOR SEVERAL MILLI-SEC
3749  013736  032762  001000  000010        BIT     #MDS,RKCS2(R2)  ;SEE IF MDS ERROR
3750  013744  001417                        BEQ     18$             ;BR IF NOT
3751  013746  112765  000101  000001        MOVB    #SELDRV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
3752  013754  011265  000016                MOV     (R2),P.CS1(R5)  ;GET RKCS1 FOR REPORT
3753  013760  004737  037000                JSR     PC,I.CSTI       ;GET OTHER RK611 REGS FOR REPORT
3754  013764  004737  030360                JSR     PC,REPSUP       ;SET UP ERROR REPORT
3755  013770  104052                        ERROR   52              ;REPORT MDS ERROR
3756  013772  052737  000200  005414        BIS     #ABORT,RECODE   ;SET ABORT FLAG
3757  014000  000137  032472                JMP     ALLTRM          ;GO ABORT TESTING
3758  014004  004737  020616          18$:  JSR     PC,INITSS       ;INIT THE S.S.
3759  014010  112765  000141  000001        MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
3760  014016  012737  020126  003046        MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADR
3761  014024  012737  000377  005466        MOV     #377,NEWON      ;INIT ONLINE INDICATOR
3762  014032  004737  027302                JSR     PC,DRVCAL       ;READ DRIVE STATUS
3763  014036  012737  030640  003046        MOV     #ERRHDL,A.ABNL  ;RESTORE ABNORMAL RETURN ADDRESS
3764  014044  022737  000377  005466        CMP     #377,NEWON      ;SEE IF THIS DRIVE EXISTS
3765  014052  001006                        BNE     16$             ;BR IF NO
3766  014054  005200                14$:    INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
3767  014056  022700  000010                CMP     #10,R0          ;SEE IF DONE CHECKING LIST
3768  014062  001302                        BNE     12$             ;BR IF NOT DONE YET
3769  014064  000137  013632                JMP     10$             ;GO BACK TO LIST SELECTED DRIVES
3770  014070  105060  005546          16$:  CLRB    SCRLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
3771  014074  000767                        BR      14$             ;CONTINUE CHECKING THE LIST
3772
3773
3774
3775
3776
3777                                  ;;*****************************************************************
3778                                  ;*TEST 1        MOUNTING OF TEST CARTRIDGE FOR PASS 1
3779                                  ;*
3780                                  ;*      THE PROGRAM TYPES "STARTING PASS 1". THEN, THE OPERATOR
3781                                  ;*      MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
3782                                  ;*      AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
3783                                  ;*      FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
3784                                  ;*      DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
3785                                  ;*      ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
3786                                  ;*      IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS.
3787                                  ;*
3788                                  ;;*****************************************************************
3789  014076  000004                  TST1:   SCOPE
3790  014100  012737  000001  001334          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3791  014106  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
3792  014112  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3793  014116  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
```

```
3794  014122  004737  021472                  JSR     PC,MTCART
3795
3796
3797
3798
3799
3800                                  ;;*****************************************************************
3801                                  ;*TEST 2        BASIC READ/WRITE DATA TEST
3802                                  ;*
3803                                  ;*      THE PROGRAM PERFORMS A WRITE AND WRITE CHECK OPERATION
3804                                  ;*      USING A "WORST CASE" DATA PATTERN, AT THE APPROPRIATE SECTOR
3805                                  ;*      FOR THIS DRIVE (FROM TABLE A) ON ALL SURFACES. THE PURPOSE
3806                                  ;*      OF THIS OPERATION IS TO VERIFY THE BASIC READ/WRITE
3807                                  ;*      CAPABILITY OF THE DRIVE ON PASS 1. THE ENTIRE SECTOR
3808                                  ;*      IS WRITTEN WITH THE REPETITION OF THE DATA PATTERN SHOWN
3809                                  ;*      IN TABLE B.
3810                                  ;*
3811                                  ;;*****************************************************************
3812  014126  000004                  TST2:   SCOPE
3813  014130  012737  000002  001334          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3814  014136  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
3815  014142  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3816  014146  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
3817  014152  112737  000001  003117          MOVB    #1,TSTING               ;ALLOW TTY ESCAPE
3818                                  ;LOAD RWBUF WITH DATA PATTERN
3819  014160  004737  023640                  JSR     PC,LODSEC
3820                                  ;SET UP PARAMETERS
3821  014164  113700  003144                  MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NUMBER
3822  014170  006300                          ASL     R0              ;CONVERT IT TO INDEX
3823  014172  006300                          ASL     R0
3824  014174  016065  005556  000002          MOV     TABLEA(R0),P.CYLN(R5) ;GET CYL # FOR THIS DRIVE
3825  014202  062700  000002                  ADD     #2,R0           ;INCREMENT INDEX
3826  014206  116065  005556  000004          MOVB    TABLEA(R0),P.SECT(R5) ;GET SECTOR # FOR THIS DRIVE
3827  014214  012765  052476  000010          MOV     #RWBUF,P.BALO(R5)       ;SET BUS ADDRESS
3828  014222  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
3829                                  ;WRITE THE DATA
3830  014230  112765  000123  000001  6$:     MOVB    #WRDATA,P.CMND(R5)      ;SET WRITE COMMAND
3831  014236  004737  027302                  JSR     PC,DRVCAL       ;WRITE THE SECTOR
3832                                  ;PERFORM WRITE CHECK OF SECTOR
3833  014242  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5)      ;SET WRITE CHECK COMMAND
3834  014250  004737  027302                  JSR     PC,DRVCAL       ;PERFORM THE WRITE CHECK
3835                                  ;INCREMENT TRACK ADDRESS
3836  014254  105265  000005                  INCB    P.TRCK(R5)      ;INCREMENT TRACK NO.
3837  014260  126527  000005  000002          CMPB    P.TRCK(R5),#2   ;SEE IF ALL SURFACES WRITTEN YET
3838  014266  101760                          BLOS    6$              ;BR IF NOT YET
3839
3840
3841
3842
3843                                  ;;*****************************************************************
3844                                  ;*TEST 3        WRITE OVRWRT AND COMPAT CYL BLOCKS FOR CURRENT DRIVE
3845                                  ;*      ALL SECTORS ARE WRITTEN FOR THE CURRENT DRIVE WITHIN THE
3846                                  ;*      CYLINDER BLOCKS IN TABLEC ON ALL SURFACES  USING A SINGLE
3847                                  ;*      REPEATED WORD OF THE PSEUDO-RANDOM DATA PATTERN IN TABLEG. LOGICAL DRIVE 0
3848                                  ;*      USES WORD 0, LOGICAL DRIVE 14 USES WORD 14, ETC.  THESE CYLINDER
3849                                  ;*      BLOCKS ARE THE OVERWRITE AND COMPATIBILITY TEST DATA TO
```

```
                                        ;*      BE USED IN PASS 2.
                                        ;;*****************************************************************
                                        TST3:   SCOPE
014270  000004
014272  012737  000003  001334          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
014300  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
014304  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
014310  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
014314  004737  022224                  JSR     PC,INTBLD       ;LOAD INITIAL VALUES INTO TABLED
014320  005000                          CLR     R0              ;INITIALIZE TABLEC INDEX TO 0
014322  016065  005716  000002  4$:     MOV     TABLEC(R0),P.CYLN(R5) ;GET START OVRWRT CYL FROM TABLEC
014330  004737  022470                  JSR     PC,WRTBLK       ;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S
014334  062700  000002                  ADD     #2,R0           ;INCREMENT TABLEC POINTER
014340  016065  005716  000002          MOV     TABLEC(R0),P.CYLN(R5) ;GET START COMPAT CYL FROM TABLEC
014346  004737  022470                  JSR     PC,WRTBLK       ;WRITE THIS COMPAT CYL BLK ON ALL SURF'S
014352  062700  000002                  ADD     #2,R0           ;INCREMENT TABLEC POINTER
014356  020027  000030                  CMP     R0,#30          ;SEE IF ON LAST CURRENT ZONE YET
014362  002003                          BGE     8$              ;BR IF YES
014364  004737  022404                  JSR     PC,ROTBLD       ;ROTATE TABLED SEVERAL SECTORS
014370  000754                          BR      4$              ;CONTINUE WRITING BLOCKS
014372  004737  022224          8$:     JSR     PC,INTBLD       ;RE-LOAD INITIAL VALUES INTO TABLED
014376  016065  005716  000002          MOV     TABLEC(R0),P.CYLN(R5) ;GET START OVRWRT CYL FROM TABLEC
014404  004737  022470                  JSR     PC,WRTBLK       ;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S


                                        ;;*****************************************************************
                                        ;*TEST 4         DISMOUNTING OF TEST CARTRIDGE IN PASS 1
                                        ;*
                                        ;*      THE PROGRAM DIRECTS THE OPERATOR TO UNLOAD THE DRIVE CURRENTLY
                                        ;*      UNDER TEST, AND TO REMOVE THE TEST PACK. THEN, THE PROGRAM
                                        ;*      DOES ONE OF 4 THINGS :
                                        ;*      (1) IF THERE IS ANOTHER DRIVE TO TEST ON THIS SUBSYSTEM, THE
                                        ;*      PROGRAM JUMPS TO TEST 1 FOR THE NEXT DRIVE.
                                        ;*      (2) IF THERE IS ONLY ONE SUBSYSTEM, (FROM 200 START), AND
                                        ;*      IF PASS 1 HAS BEEN COMPLETED ON ALL DRIVES, THE PROGRAM STARTS
                                        ;*      PASS 2 ON THE FIRST DRIVE, BY JUMPING TO TEST 5.
                                        ;*      (3) IF THERE IS ANOTHER SUBSYSTEM TO PERFORM PASS 1 ON, THE PROGRAM
                                        ;*      TELLS THE OPERATOR TO RESTART AT ADDRESS 204, AND THEN IT HALTS.
                                        ;*      (4) IF THERE ARE NO MORE DRIVES TO TEST IN PASS 1 ON ANY
                                        ;*      OTHER SUBSYSTEM, THE PROGRAM TELLS THE OPERATOR TO
                                        ;*      RESTART AT ADDRESS 220 FOR PASS 2 ON THE NEXT SUBSYSTEM, AND
                                        ;*      THEN IT HALTS.
                                        ;*
                                        ;;*****************************************************************
014410  000004                          TST4:   SCOPE
014412  012737  000004  001334          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
014420  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
014424  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
014430  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
014434  105037  003117                  CLRB    TSTING          ;DON'T ALLOW TTY ESCAPE
014440  004737  022054                  JSR     PC,DMCART       ;DIRECT OPERATOR TO DISMOUNT PACK
014444  062737  000002  005544          ADD     #2,DRVPTR       ;INCREMENT DRIVE LIST POINTER
014452  013700  005544                  MOV     DRVPTR,R0
014456  005710                          TST     (R0)            ;SEE IF ALL DONE WITH PASS 1 YET
014460  001420                          BEQ     8$              ;BR IF DONE WITH PASS 1
014462  126037  000001  005500          CMPB    1(R0),SUBSYS    ;SEE IF DONE WITH THIS SUBSYS YET
```

```
3906  014470  001004                          BNE     4$              ;BR IF DONE WITH THIS SUBSYS
3907  014472  105037  001102                  CLRB    $TSTNM          ;CLEAR TEST NUMBER
3908  014476  000137  014076                  JMP     TST1            ;DO PASS 1 ON NEXT DRIVE OF SUBSYS
3909  014502  116037  000001  010267  4$:     MOVB    1(R0),STR204+39.;PUT SUBSYS NAME INTO MSG
3910  014510  104401  010220          5$:     TYPE    ,STR204         ;TYPE RESTART MSG FOR PASS 1 ON NEXT SUBSYS
3911  014514  000000                  6$:     HALT                    ;HALT THE PROCESSOR SO THAT
3912  014516  000137  000204                  JMP     204             ;  OPERATOR CAN RESTART AT PROPER ADRS
3913  014522  012737  005502  005544  8$:     MOV     #DRVLST,DRVPTR  ;RE-INIT DRIVE LIST POINTER
3914  014530  105737  003116                  TSTB    MDFLAG          ;SEE IF 200 START
3915  014534  001007                          BNE     12$             ;BR IF NOT
3916  014536  112737  000062  010101          MOVB    #'2,SRPASS+18.  ;SET PASS NO. = 2 FOR PRINTOUT
3917  014544  104401  010057                  TYPE    ,SRPASS         ;TYPE "STARTING PASS 2"
3918  014550  000137  014574                  JMP     TST5            ;GO START PASS 2
3919  014554  112737  000101  010344  12$:    MOVB    #'A,STR220+39.  ;PUT SUBSYS NAME INTO MSG
3920  014562  104401  010275                  TYPE    ,STR220         ;TYPE RESTART MSG FOR PASS 2 ON FIRST SUBSYS
3921  014566  000000                          HALT                    ;HALT PROCESSOR SO THAT OPERATOR CAN RESTART
3922  014570  000137  000220                  JMP     220             ;  AT THE PROPER ADRS
3923
3924
3925
3926                                  ;*****************************************************************
3927                                  ;*TEST 5        MOUNTING OF TEST CARTRIDGE FOR PASS 2
3928                                  ;*
3929                                  ;*      THE PROGRAM TYPES "STARTING PASS 2", THEN, THE OPERATOR
3930                                  ;*      MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
3931                                  ;*      AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
3932                                  ;*      FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
3933                                  ;*      DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
3934                                  ;*      ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
3935                                  ;*      IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS (IF STARTED
3936                                  ;*      AT ADDRESS 204 OR 220).
3937                                  ;*
3938                                  ;*****************************************************************
3939  014574  000004          TST5:   SCOPE
3940  014576  012737  000005  001334          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3941  014604  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
3942  014610  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3943  014614  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
3944  014620  112737  000062  003145          MOVB    #'2,PASSNO      ;SET PASS NO. = 2
3945  014626  004737  021472                  JSR     PC,MTCART
3946
3947
3948
3949                                  ;*****************************************************************
3950                                  ;*TEST 6        OVERWRITE TEST
3951                                  ;*
3952                                  ;*      THE PROGRAM PROCEEDS TO TEST THIS DRIVE'S OVERWRITE CAPABILITY.
3953                                  ;*      FIRST, THE APPROPRIATE CYLINDERS IN TABLE E FOR THIS DRIVE
3954                                  ;*      ARE OVERWRITTEN, ON EACH SURFACE. THE DATA USED IS A
3955                                  ;*      REPETITION OF A SINGLE WORD OF THE PATTERN IN TABLE G,
3956                                  ;*      DRIVE 0 USES WORD 0, DRIVE 1 USES WORD 1, DRIVE 10 USES
3957                                  ;*      WORD 10, ETC.
3958                                  ;*      THEN, EACH CYLINDER OVERWRITTEN IS READ BACK BY THIS DRIVE
3959                                  ;*      WITH THE FOLLOWING RANGE OF OFFSET VALUES :  100, 200, 300,
3960                                  ;*      400,500,600,700,800,900,1000,1100,1200 MICRO-IN., IN BOTH
3961                                  ;*      THE (+) AND (-) DIRECTIONS.
```

```
                                        ;*      THE PROGRAM SCANS FOR DATA CHECK ERRORS DURING THIS READ, AND IF
                                        ;*      ONE OCCURS, THE PROGRAM DETERMINES WHICH DRIVE'S DATA HAD NOT
                                        ;*      BEEN CORRECTLY OVERWRITTEN, AND A SCORE FOR THAT DRIVE IS
                                        ;*      DECREMENTED. THEN, THE TRANSFER IS CONTINUED AT THE NEXT SECTOR,
                                        ;*      WITH THAT OFFSET VALUE. THE READS ARE DONE WITH ALL OF THE ABOVE
                                        ;*      OFFSETS APPLIED, AND A SEPARATE SCORE FOR EACH DRIVE IS KEPT,
                                        ;*      WHILE THE CURRENT DRIVE IS PERFORMING THE OVERWRITES.
                                        ;*      FOR EACH TRACK (0,1,2), SCORES ARE AVERAGED FOR EACH OFFSET
                                        ;*       DIRECTION, OVER ALL CYLINDERS TESTED.
                                        ;*
                                        ;**********************************************************************
014632  000004                  TST6:   SCOPE
014634  012737  000006  001334          MOV     #6,$TESTN       ;SET TEST NUMBER IN APT MAIL BOX
014642  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
014646  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
014652  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
014656  112737  000001  003117          MOVB    #1,TSTING       ;ALLOW TTY ESCAPE
014664  105037  003137                  CLRB    DIF100          ;CLEAR "CYL DIF = 100" FLAG
                                ;OVERWRITE THE APPROPRIATE CYLS FOR THIS DRIVE
014670  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
014676  105065  000002                  CLRB    P.CYLN(R5)      ;SET CYL = 0
014702  004737  027302                  JSR     PC,DRVCAL       ;SEEK TO CYL 0
014706  012703  000007                  MOV     #7,R3           ;SET COUNTER = 7
014712  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
014720  113700  003144                  MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NUMBER
014724  010065  000002                  MOV     R0,P.CYLN(R5)   ;SET STARTING OVERWRITE CYL FOR THIS DRIVE
014730  006300                          ASL     R0              ;GET DATA PATTERN INDEX
014732  062700  006350                  ADD     #TABLEG,R0      ;GET PATTERN WORD ADRS
014736  011037  052476                  MOV     (R0),RWBUF      ;PUT DATA WORD INTO BUFFER
014742  012765  052476  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
014750  052765  100000  000014          BIS     #DTBAII,P.PRST(R5) ;SET BUS ADRS INCREMENT INHIBIT
014756  012765  137000  000012          MOV     #-16896.,P.WC(R5) ;SET 22-SECTOR WORD COUNT
014764  105737  003124                  TSTB    FORMAT          ;DETERMINE THE FORMAT
014770  001403                          BEQ     6$              ;BR IF 22 SECTORS
014772  012765  142000  000012          MOV     #-15360.,P.WC(R5) ;SET 20-SECTOR WORD COUNT
015000  004737  026634          6$:     JSR     PC,SVPRMS       ;SAVE PARAMETERS OF TRANSFER
015004  004737  026724                  JSR     PC,TRNSFR       ;WRITE THE DATA
015010  062765  000100  000002          ADD     #100,P.CYLN(R5) ;INCR THE CYL NO.
015016  005303                          DEC     R3              ;SEE IF DONE WITH ALL CYLS YET
015020  001367                          BNE     6$              ;BR IF NOT YET
                                ;READ OVERWRITE CYLS WITH RANGE OF OFFSETS
015022  112737  000001  003133          MOVB    #1,TSTTYP       ;SET INDICATOR FOR OVRWRT TEST
015030  012700  006450                  MOV     #NOSCRO,R0      ;GET STARTING ADRS OF OVRWRT SCORES
015034  005020          10$:    CLR     (R0)+                   ;INIT SCORE TO 0
015036  020027  006750                  CMP     R0,#NOSCRO+192. ;SEE IF ALL SCORES CLEARED YET
015042  103774                          BLO     10$             ;BR IF NOT DONE YET
015044  012703  000007                  MOV     #7,R3           ;SET CYL CNTR = 7
015050  113700  003144                  MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NO.
015054  010065  000002                  MOV     R0,P.CYLN(R5)   ;GET START OVRWRT CYL FOR THIS DRIVE
015060  004737  022224          11$:    JSR     PC,INTBLD       ;INIT TABLED
015064  004737  016726          12$:    JSR     PC,REDOFS       ;DO READS WITH OFFSETS ON THIS CYL
015070  004737  017272                  JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUM
015074  062765  000100  000002          ADD     #100,P.CYLN(R5) ;INCR CYL NO.
015102  112737  000001  003137          MOVB    #1,DIF100       ;SET "CYL DIF = 100" FLAG
015110  005303                          DEC     R3              ;DECR CYL CNTR
015112  004737  022404                  JSR     PC,ROTBLD       ;ROTATE TABLED FOR NEXT CURRENT ZONE
```

```
                                                CMP     R3,#1           ;SEE IF ON LAST ZONE YET
                                                BGT     12$             ;BR IF NOT YET
                                                BEQ     11$             ;BR IF ON LAST CYL
                                                JSR     PC,AVSCOR       ;COMPUTE AVERAGE OVERWRITE SCORES
                                                CLRB    TSTTYP          ;CLEAR OVRWRT TEST INDICATOR

                                        ;*******************************************************************
                                        ;*TEST 7        DRIVE SELF-TEST
                                        ;*
                                        ;*      THE PROGRAM EVALUATES THE DRIVE'S ABILITY TO WRITE AND READ
                                        ;*      ITS OWN DATA, AT VARIOUS ANGULAR POSITIONS ON THE PACK. FIRST,
                                        ;*      SECTORS 4,11,16, AND 23 OF THE APPROPRIATE CYLINDERS SHOWN
                                        ;*      IN TABLEF FOR THIS DRIVE ARE WRITTEN WITH THE DATA PATTERN
                                        ;*      SHOWN IN TABLEB, FOR ALL SURFACES, THEN, THE SECTORS ARE READ
                                        ;*      WITH THE FOLLOWING OFFSETS APPLIED : 100,200,300,400,500,600
                                        ;*      700,800,900,1000,1100,1200 MICRO-IN.  IN (+) AND (-) DIRECTIONS.
                                        ;*      THE PROGRAM SCANS FOR DATA CHECK ERRORS DURING EACH READ, AND IT
                                        ;*      COMPUTES A SCORE WHICH IS PROPORTIONAL TO THE FAILING OFFSET
                                        ;*      MAGNITUDE.  THEN, THE SCORES FOR ALL SECTORS READ ARE AVERAGED,
                                        ;*      TO COME UP WITH A DRIVE SELF-TEST SCORE FOR EACH OFFSET
                                        ;*      DIRECTION, FOR EACH SURFACE.
                                        ;*      THIS SCORE IS SAVED FOR LATER USE, TO BECOME THE STANDARD
                                        ;*      FOR THE COMPATIBILITY DATA READS WHICH ARE TO FOLLOW.
                                        ;*
                                        ;*******************************************************************
                                TST7:   SCOPE
                                        MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
                                        JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
                                        JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
                                        JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        CLR     NSSCR0          ;INIT SELF-TEST SCORES TO 0
                                        CLR     NSSCR1
                                        CLR     NSSCR2
                                        CLR     PSSCR0
                                        CLR     PSSCR1
                                        CLR     PSSCR2
                                        CLRB    DIF100          ;CLEAR "CYL DIF = 100" FLAG
                                        MOV     #6,R3           ;SET CYL COUNTER = 6
                                        MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NO.
                                        ADD     #60,R0          ;COMPUTE STARTING CYL NO.
                                        MOV     #RWBUF,P.BAL0(R5) ;SET BUS ADRS
                                        JSR     PC,LODSEC       ;LOAD R/W BUF WITH DATA PATTERN
                                        MOV     #-256.,P.WC(R5) ;SET WORD COUNT FOR 1 SECTOR
                                3$:     MOVB    #4,P.SECT(R5)   ;SET SECTOR = 4
                                4$:     CLRB    P.TRCK(R5)      ;SET TRACK = 0
                                        CLRB    NSCOR0          ;INIT SCRATCH SCORES TO 0
                                        CLRB    NSCOR1
                                        CLRB    NSCOR2
                                        CLRB    PSCOR0
                                        CLRB    PSCOR1
                                        CLRB    PSCOR2
                                6$:     CLRB    TSTTYP          ;CLEAR OFFSET TEST INDICATOR
                                        CLR     P.CYLN(R5)      ;SET CYL = 0
                                        MOVB    #SEEK,P.CMND(R5)        ;SET SEEK COMMAND
```

```
4074  015332  004737  027302                  JSR     PC,DRVCAL       ;PERFORM SEEK TO 0
4075  015336  010065  000002                  MOV     R0,P.CYLN(R5)   ;SET CURRENT CYLINDER NUMBER
4076  015342  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
4077  015350  004737  027302                  JSR     PC,DRVCAL       ;WRITE A SECTOR
4078  015354  112737  000003  003133          MOVB    #3,TSTTYP       ;SET INDICATOR FOR SELF-TEST
4079  015362  004737  016726                  JSR     PC,REDOFS       ;READ SECTOR WITH RANGE OF OFFSETS
4080  015366  105265  000005                  INCB    P.TRCK(R5)      ;INCREMENT TRACK
4081  015372  126527  000005  000002          CMPB    P.TRCK(R5),#2   ;SEE IF ALL TRACKS DONE ON THIS CYL
4082  015400  101745                          BLOS    6$              ;BR IF NOT YET
4083  015402  004737  017272                  JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUMS
4084  015406  062765  000005  000004          ADD     #5,P.SECT(R5)   ;INCREMENT SECTOR BY 5
4085  015414  126527  000004  000023          CMPB    P.SECT(R5),#23  ;SEE IF ALL SECTORS DONE ON THIS TRACK
4086  015422  101716                          BLOS    4$              ;BR IF NOT YET
4087  015424  062700  000100                  ADD     #100,R0         ;INCR CYL NO. BY 100
4088  015430  005303                          DEC     R3              ;DECR CYL CNTR
4089  015432  001307                          BNE     3$              ;BR IF NOT DONE WITH ALL CYLS YET
4090  015434  004737  017452                  JSR     PC,AVSCOR       ;COMPUTE AVG SELF-TEST SCORES
4091  015440  105037  003133                  CLRB    TSTTYP          ;CLEAR OVRWRT TEST INDICATOR
4092
4093
4094
4095                                  ;;*****************************************************************
4096                                  ;*TEST 10       COMPATIBILITY DATA READ TEST
4097                                  ;*
4098                                  ;*      HAVING ESTABLISHED A SELF-TEST SCORE FOR THIS DRIVE, THE
4099                                  ;*      PROGRAM PROCEEDS TO PERFORM THE COMPATIBILITY DATA READS
4100                                  ;*      OF THE PATTERNS WRITTEN BY ALL THE DRIVES IN EACH CYLINDER
4101                                  ;*      BLOCK (ON EACH SURFACE). EACH COMPATIBILITY CYLINDER BLOCK
4102                                  ;*      SHOWN IN TABLE C IS READ  WITH THE FOLLOWING OFFSET VALUES :
4103                                  ;*      100,200,300,400,500,600,700,800,900,1000,1100,1200 MICRO-IN.,
4104                                  ;*      IN (+) AND (-) DIRECTIONS. THE PROGRAM SCANS FOR READ ERRORS
4105                                  ;*      DURING EACH READ, AND IF ONE OCCURS, THE PROGRAM DETERMINES
4106                                  ;*      WHICH DRIVE'S DATA WAS BEING READ AT THAT INSTANT AND A SCORE
4107                                  ;*      FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED
4108                                  ;*      AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE
4109                                  ;*      DONE WITH ALL OF THE OFFSETS APPLIED, AND A SEPARATE SCORE
4110                                  ;*      FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS READING
4111                                  ;*      THE COMPATIBILITY DATA. THEN, EACH SCORE IS APPROPRIATELY
4112                                  ;*      ADJUSTED TO REFLECT THE SELF-TEST SCORE FOR THE CURRENT DRIVE
4113                                  ;*      ON THAT SURFACE. THE SCORES ARE THEN AVERAGED OVER ALL CYLINDER
4114                                  ;*      BLOCKS. EACH SCORE IS PROPORTIONAL TO THE  CAPABILITY OF THE
4115                                  ;*      CURRENT DRIVE TO SUCCESSFULLY READ THE DATA WRITTEN BY ONE OF
4116                                  ;*      THE OTHER DRIVES, AND SCORES ARE COMPUTED SEPARATELY FOR EACH
4117                                  ;*      SURFACE (TRACK). THUS, THE SCORES REVEAL WHICH DRIVES ARE
4118                                  ;*      INVOLVED IN A SITUATION IN WHICH A PARTICULAR DRIVE HAS
4119                                  ;*      DIFFICULTY IN READING THE DATA OF ONE OR SEVERAL OTHER DRIVES.
4120                                  ;*
4121                                  ;*****************************************************************
4122  015444  000004                  TST10:  SCOPE
4123  015446  012737  000010  001334          MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4124  015454  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4125  015460  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4126  015464  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4127  015470  112737  000002  003133          MOVB    #2,TSTTYP       ;SET INDIC FOR READ TEST
4128  015476  105037  003137                  CLRB    DIF100          ;CLEAR "CYL DIF = 100" FLAG
4129  015502  012700  006750                  MOV     #NCSCRO,R0      ;GET ADRS OF COMPAT READ SCORES
```

```
 4130  015506  005020                  6$:   CLR     (R0)+           ;INIT A SCORE TO 0
 4131  015510  020027  007250                CMP     R0,#NCSCR0+192. ;SEE IF ALL SCORES CLEARED YET
 4132  015514  103774                        BLO     6$              ;BR IF NOT YET
 4133  015516  012703  000006                MOV     #6,R3           ;SET CYL BLK CNTR = 6
 4134  015522  004737  022224                JSR     PC,INTBLD       ;INIT TABLE D
 4135  015526  012701  000060                MOV     #60,R1          ;SET START CYL = 60
 4136  015532  012765  052476  000010        MOV     #RWBUF,P.BAL0(R5) ;SET BUS ADRS
 4137  015540  052765  100000  000014        BIS     #DTBAII,P.PRST(R5) ;SET BUS ADRS INCR INHIBIT
 4138  015546  012765  137000  000012        MOV     #-16896.,P.WC(R5) ;SET 22-SECTOR WORD COUNT
 4139  015554  105737  003124                TSTB    FORMAT          ;TEST THE FORMAT
 4140  015560  001403                        BEQ     10$             ;BR IF 22 SECTORS
 4141  015562  012765  142000  000012        MOV     #-15360.,P.WC(R5) ;SET 20-SECTOR WORD COUNT
 4142  015570  012700  000020          10$:  MOV     #16.,R0         ;INIT CYL CNTR = 16.
 4143  015574  010165  000002                MOV     R1,P.CYLN(R5)   ;SET CYL NO.
 4144  015600  004737  016726          12$:  JSR     PC,REDOFS       ;DO READS WITH OFSTS ON THIS CYL
 4145  015604  004737  017272                JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUM
 4146  015610  005265  000002                INC     P.CYLN(R5)      ;INCR CYL NO.
 4147  015614  005300                        DEC     R0              ;SEE IF DONE WITH THIS CYL BLK
 4148  015616  001370                        BNE     12$             ;BR IF NOT YET
 4149  015620  004737  022404                JSR     PC,ROTBLD       ;ROTATE TABLED FOR NEXT CURRENT ZONE
 4150  015624  062701  000100                ADD     #100,R1         ;INCR CYL BY 100
 4151  015630  112737  000001  003137        MOVB    #1,DIF100       ;SET "CYL DIF = 100" FLAG
 4152  015636  005303                        DEC     R3              ;SEE IF DONE WITH ALL CYL BLKS
 4153  015640  001353                        BNE     10$             ;BR IF NOT YET
 4154  015642  004737  017452                JSR     PC,AVSCOR       ;COMPUTE AVERAGES OF READ SCORES
 4155  015646  105037  003133                CLRB    TSTTYP          ;CLEAR OFST TEST INDICATOR
 4156                                ;ADJUST COMPAT READ SCORES (- OFSTS) BY SELF-TEST SCORES
 4157  015652  012701  006750                MOV     #NCSCR0,R1      ;SET COMPAT SCORE POINTER
 4158  015656  012703  007250                MOV     #NSSCR0,R3      ;SET SELF-TEST SCORE POINTER
 4159  015662  012700  005502          14$:  MOV     #DRVLST,R0      ;SET DRIVE LIST  POINTER
 4160  015666  012737  000014  001304        MOV     #12.,$TMP11     ;COMPUTE DIFFERENCE OF 12 (DEC) AND
 4161                                                                ;     SELF-TEST SCORE FOR THIS TRACK
 4162  015674  162337  001304                SUB     (R3)+,$TMP11
 4163  015700  010104                        MOV     R1,R4           ;GET READ SCORE POINTER IN R4
 4164  015702  005720                  15$:  TST     (R0)+           ;SEE IF A DRIVE LISTED HERE
 4165  015704  001417                        BEQ     18$             ;BR IF NOT
 4166  015706  013737  001304  001306        MOV     $TMP11,$TMP12   ;PUT DIFFERENCE INTO $TMP12
 4167  015714  061437  001306                ADD     (R4),$TMP12     ;ADD DIFF TO READ SCORE
 4168  015720  023727  001306  000014        CMP     $TMP12,#12.     ;SEE IF 12 EXCEEDED
 4169  015726  101403                        BLOS    16$             ;BR IF NOT
 4170  015730  012737  000014  001306        MOV     #12.,$TMP12     ;SET SCORE = 12.
 4171  015736  013724  001306          16$:  MOV     $TMP12,(R4)+    ;PUT BACK ADJUSTED SCORE
 4172  015742  000757                        BR      15$             ;BR TO CONTINUE
 4173  015744  062701  000040          18$:  ADD     #32.,R1         ;POINT TO SCORES FOR NEXT TRACK
 4174  015750  020127  007050                CMP     R1,#NCSCR0+64.  ;SEE IF SCORE TABLE EXCEEDED
 4175  015754  101742                        BLOS    14$             ;BR IF NOT YET
 4176                                ;ADJUST COMPAT READ SCORES (+ OFSTS) BY SELF-TEST SCORES
 4177  015756  012701  007110                MOV     #PCSCR0,R1      ;SET COMPAT SCORE POINTER
 4178  015762  012703  007256                MOV     #PSSCR0,R3      ;SET SELF-TEST SCORE POINTER
 4179  015766  012700  005502          24$:  MOV     #DRVLST,R0      ;SET DRIVE LIST  POINTER
 4180  015772  012737  000014  001304        MOV     #12.,$TMP11     ;COMPUTE DIFFERENCE OF 12 (DEC) AND
 4181                                                                ;     SELF-TEST SCORE FOR THIS TRACK
 4182  016000  162337  001304                SUB     (R3)+,$TMP11
 4183  016004  010104                        MOV     R1,R4           ;GET READ SCORE POINTER IN R4
 4184  016006  005720                  25$:  TST     (R0)+           ;SEE IF A DRIVE LISTED HERE
 4185  016010  001417                        BEQ     28$             ;BR IF NOT
```

```
 4186  016012  013737  001304  001306          MOV     $TMP11,$TMP12   ;PUT DIFFERENCE INTO $TMP12
 4187  016020  061437  001306                  ADD     (R4),$TMP12     ;ADD DIFF TO READ SCORE
 4188  016024  023727  001306  000014          CMP     $TMP12,#12.     ;SEE IF 12 EXCEEDED
 4189  016032  101403                          BLOS    26$             ;BR IF NOT
 4190  016034  012737  000014  001306          MOV     #12.,$TMP12     ;SET SCORE = 12.
 4191  016042  013724  001306          26$:    MOV     $TMP12,(R4)+    ;PUT BACK ADJUSTED SCORE
 4192  016046  000757                          BR      25$             ;BR TO CONTINUE
 4193  016050  062701  000040          28$:    ADD     #32.,R1         ;POINT TO SCORES FOR NEXT TRACK
 4194  016054  020127  007210                  CMP     R1,#PCSCRO+64.  ;SEE IF SCORE TABLE EXCEEDED
 4195  016060  101742                          BLOS    24$             ;BR IF NOT YET
 4196
 4197
 4198
 4199                                  ;;*****************************************************************
 4200                                  ;*TEST 11       TYPE TEST SCORES, DISMOUNT PACK, IN PASS 2
 4201                                  ;*
 4202                                  ;*      THE OVERWRITE AND COMPATIBILITY DATA READ TEST SCORES FOR THIS
 4203                                  ;*      DRIVE ARE CONVERTED AND TYPED. THEN, THE OPERATOR IS
 4204                                  ;*      DIRECTED TO UNLOAD THE DRIVE CURRENTLY UNDER TEST, AND TO REMOVE
 4205                                  ;*      THE TEST PACK. THEN, THE PROGRAM DOES 1 OF 4 THINGS :
 4206                                  ;*      (1) IF THERE IS ANOTHER DRIVE TO TEST ON THIS SUBSYS, THE
 4207                                  ;*      PROGRAM JUMPS TO TEST 5 FOR THE NEXT DRIVE.
 4208                                  ;*      (2) IF THERE IS ONLY ONE SUBSYS (FROM 200 START), AND IF
 4209                                  ;*      PASS 2 HAS BEEN COMPLETED ON ALL DRIVES, THE ENTIRE TESTING AND
 4210                                  ;*      REPORTING HAVE BEEN COMPLETED, AND THE PROGRAM TYPES A
 4211                                  ;*      COMPLETION MESSAGE, AND THEN IT RESTARTS AT ADRS 200.
 4212                                  ;*      (3) IF THERE IS ANOTHER SUBSYS TO PERFORM PASS 2 ON, THE
 4213                                  ;*      PROGRAM TELLS THE OPERATOR TO RESTART AT ADDRESS 220, AND THEN
 4214                                  ;*      IT HALTS.
 4215                                  ;*      (4) IF THERE ARE NO MORE DRIVES TO TEST ON ANY OTHER SUBSYS, THE
 4216                                  ;*      PROGRAM TYPES A COMPLETION MESSAGE AND THEN IT HALTS.
 4217                                  ;*
 4218                                  ;;*****************************************************************
 4219  016062  000004                  TST11:  SCOPE
 4220  016064  012737  000011  001334          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 4221  016072  004737  021424                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
 4222  016076  004737  020616                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
 4223  016102  004737  020422                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 4224  016106  112737  000001  003117          MOVB    #1,TSTING       ;ALLOW TTY ESCAPE
 4225                                  ;CONVERT AND TYPE TEST SCORES
 4226  016114  013700  005544                  MOV     DRVPTR,R0       ;GET DRIVE LIST POINTER
 4227  016120  112037  010513                  MOVB    (R0)+,RESLTS+20. ;GET NO. OF THIS DRIVE
 4228  016124  111037  010512                  MOVB    (R0),RESLTS+19.
 4229  016130  104401  010467                  TYPE    ,RESLTS         ;TYPE TEST RESULT HEADINGS
 4230  016134  012737  006450  001306          MOV     #NOSCRO,$TMP12  ;GET POINTER TO OVRWRT SCORES
 4231  016142  012700  000060                  MOV     #'0,R0          ;INIT TRACK NO. TO 0
 4232  016146  012701  005502          5$:     MOV     #DRVLST,R1      ;GET DRIVE LIST POINTER
 4233  016152  110037  010775                  MOVB    R0,TRKBUF+1     ;SET TRACK NO. FOR PRINTOUT
 4234  016156  013704  001306                  MOV     $TMP12,R4       ;GET SCORE POINTER
 4235  016162  004737  024722          6$:     JSR     PC,CTLOUT       ;CHECK FOR TTY ESCAPE RQST
 4236  016166  104401  010774                  TYPE    ,TRKBUF         ;TYPE TRACK (HEAD) NUMBER
 4237  016172  012103                          MOV     (R1)+,R3        ;GET A DRIVE NAME
 4238  016174  020377  167344                  CMP     R3,@DRVPTR      ;SEE IF IT IS CURRENT DRIVE
 4239  016200  001003                          BNE     8$              ;BR IF NOT
 4240  016202  104401  011005                  TYPE    ,SELF           ;TYPE "SELF"
 4241  016206  000407                          BR      10$             ;CONTINUE
```

```
 4242  016210  110337  011002          8$:   MOVB    R3,DRVBUF+2     ;GET DRIVE NO.
 4243  016214  000303                        SWAB    R3
 4244  016216  110337  011001                MOVB    R3,DRVBUF+1     ;GET SUBSYS NAME
 4245  016222  104401  011000                TYPE    ,DRVBUF         ;TYPE NO. OF DRIVE READ
 4246  016226  005037  005202          10$:  CLR     BUFF0           ;CLEAR TTY CHAR BUF
 4247  016232  005037  005204                CLR     BUFF0+2
 4248  016236  010437  001310                MOV     R4,$TMP13       ;GET SCORE POINTER
 4249  016242  012437  005202                MOV     (R4)+,BUFF0     ;PUT AN OVRWRT SCORE INTO BUF
 4250  016246  012737  000004  001304        MOV     #4,$TMP11       ;SET COUNTER = 4
 4251  016254  023727  005202  000012  11$:  CMP     BUFF0,#10.      ;SEE IF SCORE < 10.
 4252  016262  103002                        BHIS    12$             ;BR IF NOT
 4253  016264  104401  011720                TYPE    ,SPACE1         ;TYPE 1 SPACE
 4254  016270  023727  005202  000007  12$:  CMP     BUFF0,#7        ;SEE IF SCORE > 7
 4255  016276  101003                        BHI     14$             ;BR IF YES
 4256  016300  104401  011740                TYPE    ,PROMPT         ;TYPE "*"
 4257  016304  000402                        BR      16$             ;CONTINUE
 4258  016306  104401  011717          14$:  TYPE    ,SPACE2         ;TYPE 2 SPACES
 4259  016312  012746  005202          16$:  MOV     #BUFF0,-(SP)    ;GET BUF ADRS ON STACK
 4260  016316  004737  042330                JSR     PC,@#$D82D      ;CONVERT SCORE TO DEC.
 4261  016322  004737  042524                JSR     PC,@#$SUPRS     ;TYPE IT
 4262  016326  104401  011013                TYPE    ,TAB            ;TYPE A TAB
 4263  016332  062737  000140  001310        ADD     #96.,$TMP13
 4264  016340  017737  162744  005202        MOV     @$TMP13,BUFF0   ;PUT NEXT SCORE INTO BUFFER
 4265  016346  005337  001304                DEC     $TMP11          ;DECREMENT SCORE COUNTER
 4266  016352  001340                        BNE     11$             ;BR IF READ SCORE SHOULD BE TYPED
 4267  016354  104401  001325                TYPE    ,$CRLF          ;TYPE <CR>, <LF>
 4268  016360  005711                        TST     (R1)            ;SEE IF ALL SCORES TYPED FOR THIS TRACK
 4269  016362  001277                        BNE     6$              ;BR IF NOT
 4270  016364  104401  001325                TYPE    ,$CRLF          ;TYPE <CR>, <LF>
 4271  016370  005200                        INC     R0              ;INCR TRACK NO.
 4272  016372  062737  000040  001306        ADD     #32.,$TMP12     ;INCREMENT SCORE POINTER FOR NEXT TRACK
 4273  016400  020027  000062                CMP     R0,#'2          ;SEE IF ALL TRACKS TYPED YET
 4274  016404  101660                        BLOS    5$              ;BR IF NOT YET
 4275                          ;COMPUTE, CONVERT, AND TYPE TEST AVERAGES (OVER ALL SURFACES)
 4276  016406  004737  017710                JSR     PC,TRKAVG       ;GET AVGS OVER ALL TRKS
 4277  016412  013700  005544                MOV     DRVPTR,R0       ;PUT DRIVE NAME INTO BUF
 4278  016416  112037  011062                MOVB    (R0)+,BADDRV+7
 4279  016422  111037  011061                MOVB    (R0),BADDRV+6
 4280  016426  005037  001304                CLR     $TMP11          ;CLEAR SCORE INDICATOR
 4281  016432  013737  007264  005202        MOV     OVRAVG,BUFF0    ;GET OVRWRT AVG SCORE
 4282  016440  012737  010702  016454        MOV     #OAVERG,22$
 4283  016446  104401  011053          20$:  TYPE    ,BADDRV         ;TYPE "DRIVE XX"
 4284  016452  104401                        TYPE                    ;TYPE WHICH KIND OF AVERAGE
 4285  016454  000000          22$:  .WORD   0
 4286  016456  012746  005202                MOV     #BUFF0,-(SP)    ;PUT POINTER TO AVG ON STACK
 4287  016462  004737  042330                JSR     PC,@#$D82D      ;CONVERT TO DECIMAL
 4288  016466  004737  042524                JSR     PC,@#$SUPRS     ;TYPE IT
 4289  016472  104401  001325                TYPE    ,$CRLF          ;TYPE <CR>, <LF>
 4290  016476  013737  007266  005202        MOV     COMAVG,BUFF0    ;GET READ AVG SCORE
 4291  016504  012737  010721  016454        MOV     #RAVERG,22$
 4292  016512  005137  001304                COM     $TMP11          ;COMPLEMENT SCORE INDIC
 4293  016516  001353                        BNE     20$             ;BR TO TYPE READ SCORE
 4294  016520  104401  001325                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 4295                          ;DISMOUNT PACK ON THIS DRIVE
 4296  016524  105037  003117                CLRB    TSTING          ;DON'T ALLOW TTY ESCAPE
 4297  016530  004737  022054                JSR     PC,DMCART       ;DIRECT OPERATOR TO DISMOUNT PACK
```

```
4298  016534 062737 000002 005544        ADD    #2,DRVPTR       ;INCREMENT DRIVE LIST POINTER
4299  016542 013700 005544               MOV    DRVPTR,R0
4300  016546 005710                      TST    (R0)            ;SEE IF ALL DONE WITH PASS 2 YET
4301  016550 001421                      BEQ    28$             ;BR IF DONE WITH PASS 2
4302  016552 126037 000001 005500        CMPB   1(R0),SUBSYS    ;SEE IF DONE WITH THIS SUBSYS YET
4303  016560 001005                      BNE    24$             ;BR IF DONE WITH THIS SUBSYS
4304  016562 112737 000004 001102        MOVB   #4,$TSTNM       ;SET TEST NO. = 4
4305  016570 000137 014574               JMP    TST5            ;DO PASS 2 ON NEXT DRIVE OF SUBSYS
4306  016574 116037 000001 010344 24$:   MOVB   1(R0),STR220+39. ;PUT SUBSYS NAME INTO MSG
4307  016602 104401 010275               TYPE   ,STR220         ;TYPE RESTRT MSG FOR PASS 2
4308  016606 000000               26$:   HALT                   ;HALT THE PROCESSOR SO THAT OPERATOR
4309  016610 000137 000220               JMP    220             ;  CAN RESTART AT PROPER ADRS
4310  016614 104401 010740        28$:   TYPE   ,ENDTST         ;TYPE "**  END OF TESTING  **"
4311  016620 012737 005502 005544        MOV    #DRVLST,DRVPTR  ;RE-INIT DRIVE LIST POINTER
4312  016626 105737 003116               TSTB   MDFLAG          ;SEE IF 200 START
4313  016632 001002                      BNE    30$             ;BR IF NOT
4314  016634 000137 000200               JMP    200             ;RESTART AT ADRS 200
4315  016640 000000               30$:   HALT                   ;HALT THE PROCESSOR
4316  016642 000137 000204               JMP    204             ;RESTART AT ADRS 204
4317
4318
4319
4320                                     .SBTTL  END OF PASS ROUTINE
4321
4322                                     ;;*****************************************************************
4323                                     ;*INCREMENT THE PASS NUMBER ($PASS)
4324                                     ;*IF THERES A MONITOR GO TO IT
4325                                     ;*IF THERE ISN'T JUMP TO 200
4326
4327  016646                             $EOP:
4328  016646 000004                             SCOPE
4329  016650 005037 001102                      CLR    $TSTNM          ;;ZERO THE TEST NUMBER
4330  016654 005237 001336                      INC    $PASS           ;;INCREMENT THE PASS NUMBER
4331  016660 042737 100000 001336               BIC    #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
4332  016666 005327                             DEC    (PC)+           ;;LOOP?
4333  016670 000001               $EOPCT: .WORD  1
4334  016672 003013                      BGT    $DOAGN          ;;YES
4335  016674 012737                      MOV    (PC)+,@(PC)+    ;;RESTORE COUNTER
4336  016676 000001               $ENDCT: .WORD  1
4337  016700 016670                      $EOPCT
4338  016702 013700 000042       $GET42: MOV    @#42,R0         ;;GET MONITOR ADDRESS
4339  016706 001405                      BEQ    $DOAGN          ;;BRANCH IF NO MONITOR
4340  016710 000005                      RESET                  ;;CLEAR THE WORLD
4341  016712 004710               $ENDAD: JSR    PC,(R0)         ;;GO TO MONITOR
4342  016714 000240                      NOP                    ;;SAVE ROOM
4343  016716 000240                      NOP                    ;;FOR
4344  016720 000240                      NOP                    ;;ACT11
4345  016722                     $DOAGN:
4346  016722 000137 000200               JMP    @#200           ;;RETURN
4347
4348
4349
4350                                     ;;*****************************************************************
4351                                     ;*REDOFS - READ DATA WITH RANGE OF OFFSETS
4352                                     ;*THIS SUBROUTINE PERFORMS READS WITH OFFSETS LISTED IN OFSTBL.
4353                                     ;*THE OFFSETS ARE APPLIED MINIMUM TO MAXIMUM, IN BOTH DIRECTIONS,
```

```
4354                                    ;*UNTIL THERE IS NO FAILURE. EACH TIME A READ ERROR OCCURS,
4355                                    ;*THE APPROPRIATE SCORE FOR THE DRIVE INVOLVED IS INCREMENTED NO FURTHER
4356                                    ;*(IN THE SCRATCH SCORE TABLE) . EACH READ IS DONE TO A PACK
4357                                    ;*AREA, WHOSE ADDRESS IS IN THE PARAMETER BLOCK ON ENTRY.
4358                                    ;;*****************************************************************
4359  016726  104407                   REDOFS: SAVREG                  ;SAVE R0-R5
4360                                    ;INITIALIZE SCRATCH SCORES
4361  016730  122737  000003  003133            CMPB    #3,TSTTYP       ;SEE IF DRIVE SELF TEST
4362  016736  001406                            BEQ     5$              ;BR IF YES
4363  016740  012700  007270                    MOV     #NSCORO,R0      ;GET ADRS OF SCRATCH SCORE TABLE
4364  016744  105020                    4$:     CLRB    (R0)+           ;INIT A SCORE TO 0
4365  016746  020027  007430                    CMP     R0,#NSCORO+96.  ;SEE IF DONE YET
4366  016752  103774                            BLO     4$              ;BR IF NOT YET
4367                                    ;PERFORM READS WITH OFFSET
4368  016754  112765  000117  000001    5$:     MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
4369  016762  105737  003137                    TSTB    DIF100          ;SEE IF CYL DIF = 100 (OCT)
4370  016766  001412                            BEQ     9$              ;BR IF NOT
4371  016770  162765  000040  000002            SUB     #40,P.CYLN(R5)  ;SEEK HALF THE CYLS
4372  016776  004737  027302                    JSR     PC,DRVCAL
4373  017002  062765  000040  000002            ADD     #40,P.CYLN(R5)  ;SEEK THE REST OF THE WAY
4374  017010  105037  003137                    CLRB    DIF100          ;CLEAR "CYL DIF = 100 FLAG"
4375  017014  004737  027302            9$:     JSR     PC,DRVCAL       ;PERFORM SEEK
4376  017020  105037  003146                    CLRB    OFSDIR          ;CLEAR OFFSET DIRECTION FLAG
4377  017024  012703  000014                    MOV     #12.,R3         ;SET OFFSET COUNTER FOR NEG OFSTS
4378  017030  012700  000204                    MOV     #204,R0         ;INIT NEG OFST VALUE TO 100 UIN
4379  017034  105037  003147            12$:    CLRB    OFSCNT          ;INIT OFFSET NUMBER
4380  017040  122737  000003  003133    6$:     CMPB    #3,TSTTYP       ;SEE IF SELF-TEST
4381  017046  001020                            BNE     15$             ;BR IF NOT
4382  017050  116501  000005                    MOVB    P.TRCK(R5),R1   ;GET TRACK NUMBER
4383  017054  006301                            ASL     R1              ;MULT BY 16.
4384  017056  006301                            ASL     R1
4385  017060  006301                            ASL     R1
4386  017062  006301                            ASL     R1
4387  017064  062701  007270                    ADD     #NSCORO,R1      ;GET POINTER TO SCORES
4388  017070  105737  003146                    TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
4389  017074  001402                            BEQ     17$             ;BR IF NEGATIVE
4390  017076  062701  000060                    ADD     #48.,R1         ;POINT TO POSITIVE SCORES
4391  017102  012704  000001            17$:    MOV     #1,R4           ;SET COUNTER TO 1
4392  017106  000411                            BR      14$             ;CONTINUE
4393  017110  012701  007270            15$:    MOV     #NSCORO,R1      ;GET POINTER TO SCORES
4394  017114  012704  000060                    MOV     #48.,R4         ;SET COUNTER TO 48.
4395  017120  105737  003146                    TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
4396  017124  001402                            BEQ     14$             ;BR IF NEGATIVE
4397  017126  012701  007350                    MOV     #PSCORO,R1      ;GET POINTER FOR POS SCORES
4398  017132  122137  003147            14$:    CMPB    (R1)+,OFSCNT    ;SEE IF THIS IS A PERFECT SCORE YET
4399  017136  001002                            BNE     18$             ;BR IF NOT
4400  017140  105261  177777                    INCB    -1(R1)          ;INCREMENT THIS SCORE
4401  017144  005304                    18$:    DEC     R4              ;DECREMENT COUNTER
4402  017146  001371                            BNE     14$             ;BR IF NOT DONE YET
4403  017150  110065  000006            16$:    MOVB    R0,P.OFST(R5)   ;GET AN OFFSET VALUE
4404  017154  112765  000115  000001            MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET CMND
4405  017162  004737  027302                    JSR     PC,DRVCAL       ;PERFORM OFFSET
4406  017166  105237  003147                    INCB    OFSCNT          ;INCR OFFSET NUMBER
4407  017172  112765  000121  000001            MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
4408  017200  004737  026634                    JSR     PC,SVPRMS       ;SAVE PARAMETERS OF TRANSFER
4409  017204  004737  026724                    JSR     PC,TRNSFR       ;DO XFER, HANDLE DCK ERRORS
```

```
017210  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR COMMAND
017216  004737  027302                  JSR     PC,DRVCAL          ;DO A SUBSYSTEM CLEAR
017222  105065  000006                  CLRB    P.OFST(R5)         ;CLEAR OFFSET VALUE
017226  112765  000115  000001          MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
017234  004737  027302                  JSR     PC,DRVCAL          ;PERFORM OFFSET TO 0
017240  062700  000004                  ADD     #4,R0              ;INCR OFST VALUE BY 100 UIN
017244  005303                          DEC     R3                 ;DECR COUNTER
017246  001274                          BNE     6$                 ;BR IF NOT DONE IN THIS DIRECTION YET
017250  012703  000014                  MOV     #12.,R3            ;SET COUNTER FOR POS SFSTS
017254  012700  000004                  MOV     #4,R0              ;INIT POS OFST TO 100 UIN
017260  105137  003146                  COMB    OFSDIR             ;COMPLEMENT OFFSET DIRECTION FLAG
017264  001263                          BNE     12$                ;BR IF POS OFSTS NOT DONE YET
017266  104410                          RESREG                     ;RESTORE R0-R5
017270  000207                          RTS     PC                 ;RETURN


                                ;;*****************************************************************
                                ;*ADSCOR - ADD THE + AND - SCRATCH SCORES TO THE + AND - SCORE
                                ;*SUMS, FOR EITHER THE OVRWRT, SELF,  OR COMPAT READ TEST, AS DETERMINED
                                ;*BY "TSTTYP".
                                ;;*****************************************************************
017272  104407                  ADSCOR: SAVREG                     ;SAVE R0-R5
017274  122737  000003  003133          CMPB    #3,TSTTYP          ;SEE IF SELF-TEST
017302  001031                          BNE     2$                 ;BR IF NOT
017304  113700  007270                  MOVB    NSCOR0,R0          ;ADD SELF-TEST SCORES
017310  060037  007250                  ADD     R0,NSSCR0
017314  113700  007310                  MOVB    NSCOR1,R0
017320  060037  007252                  ADD     R0,NSSCR1
017324  113700  007330                  MOVB    NSCOR2,R0
017330  060037  007254                  ADD     R0,NSSCR2
017334  113700  007350                  MOVB    PSCOR0,R0
017340  060037  007256                  ADD     R0,PSSCR0
017344  113700  007370                  MOVB    PSCOR1,R0
017350  060037  007260                  ADD     R0,PSSCR1
017354  113700  007410                  MOVB    PSCOR2,R0
017360  060037  007262                  ADD     R0,PSSCR2
017364  000430                          BR      8$                 ;EXIT
017366  012700  006450          2$:     MOV     #NOSCR0,R0         ;GET POINTER TO (-) OVRWRT SCORES
017372  012705  006610                  MOV     #POSCR0,R5         ;GET POINTER TO (+) OVRWRT SCORES
017376  122737  000001  003133          CMPB    #1,TSTTYP          ;SEE IF READING OVRWRT DATA
017404  001404                          BEQ     4$                 ;BR IF YES
017406  012700  006750                  MOV     #NCSCR0,R0         ;GET POINTER TO (-) COMPAT DATA SCORES
017412  012705  007110                  MOV     #PCSCR0,R5         ;GET POINTER TO (+) COMPAT DATA SCORES
017416  012704  000060          4$:     MOV     #48.,R4            ;INIT COUNTER TO 48.
017422  012701  007270                  MOV     #NSCOR0,R1         ;GET POINTER TO NEG SCRATCH SCORES
017426  012703  007350                  MOV     #PSCOR0,R3         ;GET POINTER TO POS SCRATCH SCORES
017432  112102                  6$:     MOVB    (R1)+,R2           ;GET A SCRATCH NEG SCORE
017434  060220                          ADD     R2,(R0)+           ;ADD IT TO (-) SCORE SUM
017436  112302                          MOVB    (R3)+,R2           ;GET PNTR TO POS SCRATCH SCORES
017440  060225                          ADD     R2,(R5)+           ;ADD IT TO (+) SCORE SUM
017442  005304                          DEC     R4                 ;DECREMENT COUNTER
017444  001372                          BNE     6$                 ;BR IF NOT DONE YET
017446  104410                  8$:     RESREG                     ;RESTORE R0-R5
017450  000207                          RTS     PC                 ;RETURN
```

```
                                        ;;*****************************************************************
                                        ;*AVSCOR - COMPUTE THE + AND - AVERAGE SCORES FOR THE OVRWRT, COMPATIBILITY
                                        ;*DATA TEST, OR SELF-TEST, AS DETERMINED BY "TSITYP". THE SUMS IN THE APPROPRIATE
                                        ;*SCORE TABLES ARE EACH DIVIDED BY THE NUMBER OF CYLINDERS READ, TO
                                        ;*COMPUTE THE AVERAGES.
                                        ;;*****************************************************************
017452  104407                  AVSCOR: SAVREG                  ;SAVE R0-R5
017454  122737  000003  003133          CMPB    #3,TSTTYP       ;SEE IF SELF-TEST
017462  001014                          BNE     2$              ;BR IF NOT
017464  012737  007250  001304          MOV     #NSSCRO,$TMP11  ;GET ADRS OF (-) SELF-TEST SCORES
017472  012737  007256  001310          MOV     #PSSCRO,$TMP13  ;GET ADRS OF (+) SELF-TEST SCORES
017500  012705  000030                  MOV     #24.,R5         ;SET DIVISOR = 24.
017504  012737  000003  001306          MOV     #3,$TMP12       ;SET COUNTER = 3
017512  000427                          BR      12$             ;CONTINUE
017514  012737  006450  001304  2$:     MOV     #NOSCRO,$TMP11  ;GET POINTER TO (-) OVRWRT SCORES
017522  012737  006610  001310          MOV     #POSCRO,$TMP13  ;GET POINTER TO (+) OVRWRT SCORES
017530  012705  000007                  MOV     #7.,R5          ;SET DIVISOR = 7.
017534  122737  000001  003133          CMPB    #1,TSTTYP       ;SEE IF READING OVRWRT DATA
017542  001410                          BEQ     4$              ;BR IF YES
017544  012737  006750  001304          MOV     #NCSCRO,$TMP11  ;GET POINTER TO (-) COMPAT DATA SCORES
017552  012737  007110  001310          MOV     #PCSCRO,$TMP13  ;GET POINTER TO (+) COMPAT DATA SCORES
017560  012705  000140                  MOV     #96.,R5         ;SET DIVISOR = 96.
017564  012737  000060  001306  4$:     MOV     #48.,$TMP12     ;SET COUNTER = 48.
017572  105037  003146          12$:    CLRB    OFSDIR          ;INIT FLAG FOR (-) DIRECTION
017576  005000                  6$:     CLR     R0              ;CLEAR HI BITS OF DIVIDEND AND DIVISOR
017600  005001                          CLR     R1
017602  005002                          CLR     R2
017604  005004                          CLR     R4
017606  017703  161472                  MOV     @$TMP11,R3      ;SET DIVIDEND = (-) SUM
017612  105737  003146                  TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
017616  001402                          BEQ     10$
017620  017703  161464                  MOV     @$TMP13,R3      ;SET DIVIDEND = (+) SUM
017624  004737  042032          10$:    JSR     PC,M.DPID       ;DIVIDE TO GET AVERAGES
017630  010500                          MOV     R5,R0           ;GET DIVISOR
017632  006200                          ASR     R0              ;DIVIDE IT BY 2
017634  020100                          CMP     R1,R0           ;SEE IF REMNDR > HALF DIVISOR
017636  103401                          BLO     8$              ;BR IF NOT
017640  005203                          INC     R3              ;ADD 1 TO QUOTIENT
017642  105137  003146          8$:     COMB    OFSDIR          ;COMPL DIR FLAG
017646  001406                          BEQ     14$             ;BR FOR (+)
017650  010377  161430                  MOV     R3,@$TMP11      ;PUT (-) AVG INTO TABLE
017654  062737  000002  001304          ADD     #2,$TMP11
017662  000745                          BR      6$
017664  010377  161420          14$:    MOV     R3,@$TMP13      ;PUT (+) AVG INTO TABLE
017670  062737  000002  001310          ADD     #2,$TMP13
017676  005337  001306                  DEC     $TMP12          ;DECREMENT COUNTER
017702  001333                          BNE     12$             ;BR IF NOT DONE YET
017704  104410                          RESREG                  ;RESTORE R0-R5
017706  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        ;*TRKAVG - COMPUTE AVERAGES OF TEST SCORES OVER ALL SURFACES
```

```
                                ;*THIS SUBROUTINE COMPUTES THE AVERAGE OF THE OVERWRITE TEST SCORES
                                ;*OVER ALL TRACKS (SURFACES) AND PLACES THE AVERAGE IN OVRAVG.
                                ;*THEN, IT COMPUTES THE AVERAGE OF THE COMPATIBILITY DATA READ SCORES
                                ;*OVER ALL TRACKS AND PLACES THE AVERAGE IN COMAVG.
                                ;*****************************************************************
017710  104407                  TRKAVG: SAVREG                  ;SAVE R0-R5
                                ;TAKE SUMS OF OVERWRITE AND COMPAT READ TEST SCORES OVER ALL TRACKS
017712  012700  005502                  MOV     #DRVLST,R0      ;INIT DRIVE LIST POINTER
017716  005001                          CLR     R1              ;INIT DRIVE COUNTER
017720  005037  007264                  CLR     OVRAVG          ;INIT SUMS TO 0
017724  005037  007266                  CLR     COMAVG
017730  012702  006450                  MOV     #NOSCRO,R2      ;SET POINTER TO OVRWRT SCORES
017734  012703  006750                  MOV     #NCSCRO,R3      ;SET POINTER TO READ SCORES
017740  005720                  6$:     TST     (R0)+           ;COMPUTE TOTAL NUMBER OF DRIVES
017742  001402                          BEQ     8$              ;    IN R1
017744  005201                          INC     R1
017746  000774                          BR      6$
017750  010100                  8$:     MOV     R1,R0           ;PUT NO. OF DRIVES IN R0
017752  010204                          MOV     R2,R4           ;SET POINTERS TO SCORES FOR THIS TRACK
017754  010305                          MOV     R3,R5
017756  062437  007264          9$:     ADD     (R4)+,OVRAVG    ;ADD SCORES TO SUMS
017762  066437  000136  007264          ADD     94.(R4),OVRAVG
017770  062537  007266                  ADD     (R5)+,COMAVG
017774  066537  000136  007266          ADD     94.(R5),COMAVG
020002  005300                          DEC     R0              ;SEE IF ALL SCORES ADDED FOR THIS TRACK
020004  001364                          BNE     9$              ;BR IF NOT YET
020006  062702  000040                  ADD     #32.,R2         ;POINT TO SCORES FOR NEXT TRACK
020012  062703  000040                  ADD     #32.,R3
020016  020227  006550                  CMP     R2,#NOSCRO+64.  ;SEE IF ALL TRACKS DONE YET
020022  101752                          BLOS    8$              ;BR IF NOT YET
                                ;DIVIDE TO GET AVERAGES
020024  005004                          CLR     R4              ;GET DIVISOR IN R4-R5
020026  005005                          CLR     R5
020030  060105                          ADD     R1,R5
020032  060105                          ADD     R1,R5
020034  060105                          ADD     R1,R5
020036  006305                          ASL     R5
020040  012737  007264  001304          MOV     #OVRAVG,$TMP11  ;SET POINTER TO OVRWRT SCORE SUM
020046  005037  001306                  CLR     $TMP12          ;CLEAR SCORE INDICATOR
020052  005000                  10$:    CLR     R0              ;GET DIVIDEND IN R0-R1-R2-R3
020054  005001                          CLR     R1
020056  005002                          CLR     R2
020060  017703  161220                  MOV     @$TMP11,R3
020064  004737  042032                  JSR     PC,M.DPID       ;DIVIDE TO GET AVERAGE
020070  010500                          MOV     R5,R0           ;GET DIVISOR
020072  006200                          ASR     R0              ;DIVIDE IT BY 2
020074  020100                          CMP     R1,R0           ;SEE IF REMNDR > HALF DIVISOR
020076  103401                          BLO     12$             ;BR IF NOT
020100  005203                          INC     R3              ;ADD 1 TO QUOTIENT
020102  010377  161176          12$:    MOV     R3,@$TMP11      ;STORE AVERAGE SCORE
020106  012737  007266  001304          MOV     #COMAVG,$TMP11  ;SET POINTER FOR READ SCORE SUM
020114  005137  001306                  COM     $TMP12          ;COMPLEMENT SCORE INDICATOR
020120  001354                          BNE     10$             ;BR TO COMPUTE READ SCORE AVERAGE
020122  104410                          RESREG                  ;RESTORE R0-R5
020124  000207                          RTS     PC              ;RETURN
```

```
4578
4579
4580                                    ;;*****************************************************************
4581                                    ;*NEDHDL - NON-EXISTENT DRIVE HANDLER
4582                                    ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
4583                                    ;*  WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
4584                                    ;*    (NED = NON-EXISTENT DRIVE)
4585                                    ;;*****************************************************************
4586  020126  032765  010000  000020  NEDHDL: BIT     #NED,P.CS2(R5)  ;SEE IF NED ON DRIVE SELECT
4587  020134  001002                          BNE     1$              ;BR IF NED SET
4588  020136  000137  030640                  JMP     ERRHDL          ;GO HANDLE  OTHER ERROR
4589  020142  010046                  1$:     MOV     R0,-(SP)        ;SAVE R0,R1
4590  020144  010146                          MOV     R1,-(SP)
4591  020146  012700  000001                  MOV     #1,R0           ;SET BIT POINTER
4592  020152  005001                          CLR     R1              ;CLEAR COUNTER
4593  020154  120137  005420          2$:     CMPB    R1,DRIVE        ;SEE IF R1 = CURRENT DRIVE NUMBER
4594  020160  001403                          BEQ     3$              ;BR IF  =
4595  020162  005201                          INC     R1              ;INCREMENT COUNTER
4596  020164  006300                          ASL     R0              ;SHIFT BIT POINTER
4597  020166  000772                          BR      2$              ;TRY AGAIN
4598  020170  040037  005466          3$:     BIC     R0,NEWON        ;CLEAR ONLINE BIT FOR  THIS DRIVE
4599  020174  012601                          MOV     (SP)+,R1        ;RESTORE R0,R1
4600  020176  012600                          MOV     (SP)+,R0
4601  020200  000137  033110                  JMP     RETNML          ;TAKE  NORMAL  RETURN
4602
4603
4604
4605
4606
4607                                    ;;*****************************************************************
4608                                    ;*DCKHDL - DATA CHECK ERROR HANDLER
4609                                    ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER, WHICH IS USED
4610                                    ;*WHEN DCK ERROR IS EXPECTED (AND VALID).
4611                                    ;*IF A HEADER VRC ERROR OCCURS, IT IS HANDLED, ALSO.
4612                                    ;;*****************************************************************
4613  020204  005037  005414  000000  DCKHDL: CLR     RECODE          ;CLEAR ERROR FLAGS
4614  020210  032765  000400  000034          BIT     #HVRC,P.ER(R5)  ;SEE IF HEADER VRC ERROR
4615  020216  001404                          BEQ     4$              ;BR IF NOT
4616  020220  052737  000004  005414          BIS     #HVRCER,RECODE  ;SET HVRC ERROR FLAG
4617  020226  000411                          BR      8$              ;EXIT
4618  020230  032765  100000  000034  4$:     BIT     #DCK,P.ER(R5)   ;SEE IF DCK ERROR OCCURRED
4619  020236  001002                          BNE     6$              ;BR IF YES
4620  020240  000137  030640                  JMP     ERRHDL          ;GO HANDLE OTHER ERROR
4621  020244  052737  000020  005414  6$:     BIS     #DCKERR,RECODE  ;SET DCK ERROR FLAG
4622  020252  105237  003134          8$:     INCB    DCEFLG          ;INCREMENT DCK ERROR FLAG
4623  020256  000137  033110                  JMP     RETNML          ;TAKE NORMAL RETURN FROM DRIVER
4624
4625
4626
4627                                    ;;*****************************************************************
4628                                    .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
4629                                    ;;*****************************************************************
4630
4631  020262  010146                  KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
4632  020264  017701  160656                  MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
4633  020270  042701  177600                  BIC     #177600,R1      ;MASK OUT UNUSED BITS
```

```
4634  020274  120127  000172          CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
4635  020300  003005                  BGT     20$             ;BR IF NOT
4636  020302  120127  000141          CMPB    R1,#141
4637  020306  002402                  BLT     20$             ;BR IF NOT
4638  020310  042701  000040          BIC     #BIT5,R1        ;MAKE IT UPPER CASE
4639  020314  010137  005432  20$:    MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
4640  020320  122701  000003          CMPB    #003,R1         ;SEE IF (↑C) TYPED
4641  020324  001007                  BNE     2$              ;BR IF NOT (↑C)
4642  020326  104401  011652          TYPE    ,CNTRLC         ;ECHO (↑C)
4643  020332  042777  000100  160604  1$:     BIC     #BIT6,@STKS     ;CLEAR KBD INTERRUPT ENABLE BIT
4644  020340  012601                  MOV     (SP)+,R1        ;RESTORE R1
4645  020342  000002                  RTI                     ;RETURN
4646  020344  122701  000032          CMPB    #032,R1         ;SEE IF (↑Z) TYPED
4647  020350  001003                  BNE     3$              ;BR IF NOT (↑Z)
4648  020352  104401  011657          TYPE    ,CNTRLZ         ;ECHO (↑Z)
4649  020356  000765                  BR      1$              ;RETURN
4650  020360  122701  000022          CMPB    #022,R1         ;SEE IF (↑R) TYPED
4651  020364  001003                  BNE     4$              ;BR IF NOT (↑R)
4652  020366  104401  011664          TYPE    ,CNTRLR         ;ECHO (↑R)
4653  020372  000757                  BR      1$              ;RETURN
4654  020374  122701  000007          CMPB    #007,R1         ;SEE IF (↑G) TYPED
4655  020400  001003                  BNE     5$              ;BR IF NOT (↑G)
4656  020402  104401  011676          TYPE    ,CNTRLG         ;ECHO (↑G)
4657  020406  000751                  BR      1$              ;RETURN
4658  020410  104401  005432  5$:     TYPE    ,INTCHR         ;ECHO INPUT
4659  020414  104401  001325          TYPE    ,$CRLF          ;DO <CR> AND <LF>
4660  020420  000744                  BR      1$              ;RETURN
4661
4662
4663
4664                                  ;*****************************************************************
4665                                  .SBTTL  PREPKB - PREPARE FOR KEYBOARD INPUT
4666                                  ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
4667                                  ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
4668                                  ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
4669                                  ;*ENABLES KBD INTERRUPT.
4670                                  ;* CALL:
4671                                  ;*      JSR     PC,PREPKB
4672                                  ;*****************************************************************
4673
4674  020422  005077  160520  PREPKB: CLR     @STKB           ;CLEAR KBD BUFFER AND DONE BIT
4675  020426  005037  005432          CLR     INTCHR          ;CLEAR TTY INPUT WORD
4676  020432  052777  000100  160504  BIS     #BIT6,@STKS     ;ENABLE KBD INTERRUPT
4677  020440  000207                  RTS     PC              ;RETURN
4678
4679
4680
4681                                  ;*****************************************************************
4682                                  .SBTTL  ECOBAD - ECHO BAD TTY INPUT
4683                                  ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
4684                                  ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
4685                                  ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
4686                                  ;*AND <LF> ARE DONE.
4687                                  ;* CALL -      JSR     PC,ECOBAD
4688                                  ;*****************************************************************
4689
```

```
                                ECOBAD: TYPE    ,INTCHR         ;ECHO BAD CHARACTER
                                        TYPE    ,$QUES          ;TYPE <?> AND <CR>, <LF>
                                        RTS     PC              ;RETURN
;;*****************************************************************
;* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
;*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
;*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
;*TOP OF STACK.
;;*****************************************************************
020454  016646  000002  GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
020460  104403                  TYPOS                   ;TYPE IT
020462     006                  .BYTE   6               ;SIX DIGITS
020463     000                  .BYTE   0               ;SUPPRESS LEADING ZEROS
020464  104401  011024          TYPE    ,NEWMSG         ;TYPE "  NEW = "
020470  004737  023006          JSR     PC,RDCHRS       ;READ NEW VALUE FROM KBD
020474  020526                  7$                      ;(↑C) RETURN ADDRESS
020476  020526                  7$                      ;(↑Z) RETURN ADDRESS
020500  020526                  7$                      ;(↑U) OR ERROR RETURN ADDRESS
020502  005700                  TST     R0              ;SEE IF ANY CHARS TYPED
020504  001001                  BNE     4$              ;BR IF YES
020506  000207                  RTS     PC              ;RETURN - OLD VALUE UNCHANGED
020510  020027  000006  4$:     CMP     R0,#6           ;SEE IF > 6 CHARS TYPED
020514  003407                  BLE     8$              ;BR IF NOT BAD
020516  104401  005202  6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
020522  104401  001324          TYPE    ,$QUES
020526  162716  000010  7$:     SUB     #10,(SP)        ;FIX ERROR RETURN PC
020532  000207                  RTS     PC              ;ERROR RETURN
020534  012746  005202  8$:     MOV     #BUFFO,-(SP)    ;PUT POINTER TO CHARS ON STACK
020540  004737  041252          JSR     PC,OCTBIN       ;CONVERT DIGITS TO BINARY
020544  020516                  6$                      ;ERROR RETURN ADDRESS
020546  012600                  MOV     (SP)+,R0        ;GET NEW BINARY VALUE
020550  005737  041404          TST     $HIOCT          ;SEE IF HI BITS ARE 0
020554  001360                  BNE     6$              ;BR IF NOT
020556  010066  000002          MOV     R0,2(SP)        ;PUT NEW VALUE ON STACK
020562  000207                  RTS     PC              ;RETURN
;;*****************************************************************
        .SBTTL  GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION
;*THIS SUBROUTINE ALLOWS THE CONTENTS OF THE SOFTWARE SWITCH
;*REGISTER TO BE MODIFIED BY TTY INPUT. THE SUBROUTINE TYPES
;* "SWR = XXXXXX  NEW = ", AND WAITS FOR A NEW OCTAL VALUE
;*OF UP TO SIX DIGITS TO BE TYPED.
;;*****************************************************************
020564  022737  000176  001140  GTSWRG: CMP     #SWREG,SWR      ;SEE IF SOFTWARE SWR SELECTED
020572  001010                  BNE     6$              ;BR IF NOT
020574  013746  000176          MOV     SWREG,-(SP)     ;PUT OLD VALUE ON STACK
020600  104401  011015          TYPE    ,SWRMSG         ;TYPE "SWR = "
020604  004737  020454          JSR     PC,GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
```

```
020610  012637  000176                  MOV     (SP)+,SWREG     ;STORE NEW VALUE
020614  000207                  6$:     RTS     PC              ;RETURN


                        ;;*****************************************************************
                        .SBTTL  INITSS - INITIALIZE SUBSYSTEM
                        ;*
                        ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
                        ;*AND DOES A SUBSYSTEM CLEAR.
                        ;* USES -   R2,R5
                        ;* CALL:
                        ;*              JSR     PC,INITSS
                        ;;*****************************************************************

020616  012737  030640  003046  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
020624  012737  027426  003044          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
020632  013702  003036                  MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
020636  012705  002630                  MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
020642  105037  003140                  CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
020646  004737  020710                  JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
020652  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
020660  004737  027302                  JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
020664  113765  005420  000000          MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
020672  113737  005420  002714          MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
020700  113765  003124  000007          MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
020706  000207                          RTS     PC              ;SUBROUTINE EXIT


                        ;;*****************************************************************
                        .SBTTL  CLRPRM - CLEAR DRIVER INPUT PARAMETERS
                        ;*THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
                        ;*PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
                        ;* CALL -       JSR     PC,CLRPRM
                        ;;*****************************************************************

020710  010046          CLRPRM: MOV     R0,-(SP)        ;SAVE R0
020712  010546                  MOV     R5,-(SP)        ;SAVE R5
020714  010500                  MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
020716  062705  000016          ADD     #P.CS1,R5       ;COMPUTE LIMIT ADDRESS
020722  005020          1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
020724  020005                  CMP     R0,R5           ;SEE IF DONE YET
020726  001375                  BNE     1$              ;BR IF  NOT DONE  YET
020730  012605                  MOV     (SP)+,R5        ;RESTORE R5
020732  012600                  MOV     (SP)+,R0        ;RESTORE R0
020734  000207                  RTS     PC              ;RETURN


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

```
                                        .SBTTL  SCNDRV - SCAN DRIVE FOR STATUS
                                        ;*THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
                                        ;*THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
                                        ;*AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
                                        ;*OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
                                        ;*MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
                                        ;*A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
                                        ;*AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
                                        ;*MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
                                        ;* CALL -    JSR     PC,SCNDRV
                                        ;*              <ERROR RETURN ADDRESS>
                                        ;*
                                        ;;*****************************************************************

020736  004737  020616          SCNDRV: JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
020742  042712  000100          3$:     BIC     #IE,(R2)        ;DISABLE RK06 INTERRUPT
020746  113762  005420  000010          MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
020754  012712  000001                  MOV     #BIT0,(R2)      ;SELECT THE DRIVE
020760  005037  005434                  CLR     SCRACH          ;CLEAR STALL COUNTER
020764  005237  005434          15$:    INC     SCRACH          ;INCREMENT STALL COUNTER
020770  001375                  BNE     15$             ;STALL FOR SEVERAL MILLI-SEC
020772  032762  001000  000010          BIT     #MDS,RKCS2(R2)  ;SEE IF MDS ERROR
021000  001417                  BEQ     18$             ;BR IF NOT
021002  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
021010  011265  000016                  MOV     (R2),P.CS1(R5)  ;GET RKCS1 FOR REPORT
021014  004737  037000                  JSR     PC,I.CST1       ;GET OTHER RK611 REGS FOR REPORT
021020  004737  030360                  JSR     PC,REPSUP       ;SET UP ERROR REPORT
021024  104052                  ERROR   52              ;REPORT MDS ERROR
021026  052737  000200  005414          BIS     #ABORT,RECODE   ;SET ABORT FLAG
021034  000137  032472                  JMP     ALLTRM          ;GO ABORT TESTING
021040  004737  020616          18$:    JSR     PC,INITSS       ;INIT THE S.S.
021044  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
021052  012737  020126  003046          MOV     #NEDHDL,@#ABNL  ;SET NED ABNORMAL RETURN ADDRESS
021060  012737  000377  005466          MOV     #377,NEWON      ;INIT. ON-LINE INDICATOR
021066  004737  027302                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
021072  112737  000040  011062          MOVB    #40,BADDRV+7    ;PUT A SPACE IN MSG BUF
                        ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
021100  012737  030640  003046          MOV     #ERRHDL,@#ABNL  ;RESTORE ABNL RETURN ADDRESS
021106  022737  000377  005466          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
021114  001420                  BEQ     4$              ;BR IF NED NOT SET
021116  113737  005420  011061          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
021124  152737  000060  011061          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
021132  104401  011053                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
021136  104401  011064                  TYPE    ,NXDRIV         ;TYPE "NON-EXISTENT"
                        ;SERVICE ERRORS HERE
021142  042762  000100  000000  2$:     BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
021150  017616  000000                  MOV     @(SP),(SP)      ;SET UP ERROR RETURN ADDRESS
021154  000207                  RTS     PC              ;ERROR RETURN
                        ;SEE IF DRIVE IS READY
021156  032765  000200  000040  4$:     BIT     #S.DRY,P.ADD(R5) ;TEST FOR DRIVE READY
021164  001013                  BNE     6$              ;BR IF DRIVE IS READY
021166  113737  005420  011061          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
021174  152737  000060  011061          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
021202  104401  011053                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
021206  104401  011103                  TYPE    ,NTREDY         ;TYPE "NOT READY"
021212  000753                  BR      2$              ;TAKE ERROR EXIT
```

```
                                ;SEE IF DRIVE IS WRITE ENABLED
021214  032765  004000  000040  6$:     BIT     #S.WRL,P.ADD(R5) ;SEE IF WRITE LOCK SET
021222  001413                          BEQ     8$              ;BR IF WRITE LOCK NOT SET
021224  113737  005420  011061          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
021232  152737  000060  011061          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
021240  104401  011053                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
021244  104401  011117                  TYPE    ,WRTLOK         ;TYPE "WRITE-LOCKED"
021250  000734                          BR      2$              ;TAKE ERROR EXIT
                                ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
021252  112765  000103  000001  8$:     MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
021260  004737  027302                  JSR     PC,DRVCAL       ;SET VOLUME VALID
021264  112765  000113  000001          MOVB    #RECAL,P.CMND(R5)       ;SET RECAL CMND
021272  004737  027302                  JSR     PC,DRVCAL       ;RECAL THE DRIVE
021276  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
021304  012765  000632  000002          MOV     #LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)
021312  112765  000002  000005          MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
021320  012765  052476  000010          MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
021326  012765  177774  000012          MOV     #-4,P.WC(R5)    ;READ 4 WORDS
021334  105065  000007                  CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
021340  004737  027302                  JSR     PC,DRVCAL       ;READ 4 WORDS OF BAD SECTOR FILE
021344  032737  100000  005414          BIT     #ANYDER,RECODE  ;SEE IF DATA ERROR
021352  001402                          BEQ     10$             ;BR IF OK
021354  104401  011606                  TYPE    ,BAD632         ;TYPE READ ERROR MESSAGE
021360  022737  177777  052504  10$:     CMP     #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
021366  001013                          BNE     12$             ;BR IF NOT ALL 1'S
021370  113737  005420  011061          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
021376  152737  000060  011061          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
021404  104401  011053                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
021410  104401  011136                  TYPE    ,ALNPAK         ;TYPE "LOADED WITH ALIGN PACK"
021414  000652                          BR      2$              ;TAKE ERROR EXIT
                                ;ERROR FREE RETURN
021416  062716  000002          12$:    ADD     #2,(SP)         ;FIX UP RETURN PC
021422  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*SETTUP - SET UP FOR LOOP ON ERROR
                                ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
                                ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
                                ;;*****************************************************************
021424  011637  001076          SETTUP: MOV     (SP),@#STACK-2  ;MOVE RETURN PC ON STACK
021430  012706  001076                  MOV     #STACK-2,SP     ;RE-INIT THE STACK POINTER
021434  005037  005414                  CLR     RECODE          ;CLEAR ERROR RECOVERY FLAGS
021440  105037  003125                  CLRB    ERRCNT          ;CLEAR RETRY ERROR COUNT
021444  012705  002630                  MOV     #PARMO,R5       ;SET PARAM BLK ADRS
021450  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
021456  004737  027302                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
021462  012737  000000  177776          MOV     #PRO,@#PS       ;RE-ESTABLISH PRIORITY 0
021470  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  MTCART - MOUNT TEST CARTRIDGE, CHECK DRIVE STATUS
```

```
                                        ;*THIS SUBROUTINE PERFORMS THE FOLLOWING FUNCTIONS :
                                        ;*IT TYPES "STARTING PASS X", DIRECTS THE OPERATOR TO MOUNT THE
                                        ;*TEST CARTRIDGE, CHECKS THE DRIVE STATUS, DETERMINES THE PACK FORMAT,
                                        ;*READS THE BAD SECTOR FILES, AND TYPES THE DRIVE AND CART. SERIAL
                                        ;*NUMBERS.
                                        ;;*****************************************************************
021472                                  MTCART:
021472  013700  005544                         MOV     DRVPTR,R0       ;GET POINTER TO FIRST DRIVE FOR THIS SUBSYS
021476  112037  010136                         MOVB    (R0)+,MNTPAK+22.;PUT DRIVE NAME INTO MSG
021502  111037  010135                         MOVB    (R0),MNTPAK+21.
021506  162700  005502                         SUB     #DRVLST,R0      ;SUBTRACT LIST ADRS FROM POINTER
021512  006200                               ASR     R0              ;DIVIDE BY TWO TO GET DRIVE NO.
021514  110037  003144                         MOVB    R0,LOGDRV       ;STORE LOGICAL DRIVE NUMBER
021520  004737  020616                         JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
021524  042762  000100  000000                 BIC     #IE.RKCS1(R2)   ;DISABLE RK06 INTERRUPT FOR MAN. INTERVENTION
021532  104401  010110                 10$:    TYPE    ,MNTPAK         ;TYPE MOUNT PACK MSG
021536  104401  010352                 5$:     TYPE    ,TYPRWN         ;TYPE "TYPE R<CR> WHEN "
021542  104401  010373                         TYPE    ,DRIRDY         ;TYPE "DRIVE READY :"
021546  004737  023006                         JSR     PC,RDCHRS       ;WAIT FOR R<CR> RESPONSE
021552  021576                               12$                     ;(↑C) RETURN ADDRESS
021554  021620                               16$                     ;(↑Z) RETURN ADDRESS
021556  021536                               5$                      ;(↑U) RETURN ADDRESS
021560  005700                               TST     R0              ;SEE IF NULL INPUT
021562  001025                               BNE     8$              ;BR IF NOT NULL
021564  104401  005202                 6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
021570  104401  001324                         TYPE    ,SQUES          ;TYPE <?> AND <CR>,<LF>
021574  000760                               BR      5$              ;GO ASK AGAIN
021576  012706  001100                 12$:    MOV     #STACK,SP       ;RE-INIT THE STACK
021602  105737  003116                         TSTB    MDFLAG          ;SEE IF 200 START
021606  001002                               BNE     14$             ;BR IF NOT
021610  000137  011746                         JMP     DFSTRT          ;RESTART PROGRAM - 200 START
021614  000137  012546                 14$:    JMP     ALLSYS          ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
021620  105737  003116                 16$:    TSTB    MDFLAG          ;SEE IF 200 START
021624  001744                               BEQ     5$              ;BR IF YES, TO ASK AGAIN
021626  012706  001100                         MOV     #STACK,SP       ;RE-INIT THE STACK
021632  000137  013212                         JMP     ASKSYS          ;GO ASK FOR SUBSYS TO TEST AGAIN
021636  023727  005202  000122         8$:     CMP     BUFFO,#'R       ;SEE IF R<CR> TYPED
021644  001347                               BNE     6$              ;BR IF NOT
021646  004737  020616                         JSR     PC,INITSS       ;INITIALIZE S.S.
                                        ;CHECK STATUS OF DRIVE TO BE TESTED NEXT
021652  117737  163666  005420                 MOVB    @DRVPTR,DRIVE   ;SET DRIVE NUMBER
021660  142737  000060  005420                 BICB    #'0,DRIVE       ;STRIP THE ASCII
021666  004737  020736                         JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE -
021672  021532                               10$                     ;  IF NOT VALID, TYPE MSG AND RETURN
                                                                     ;  TO WAIT FOR R<CR> AGAIN
                                        ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
021674  004737  020616                         JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
021700  112765  000125  000001                 MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
021706  004737  027302                         JSR     PC,DRVCAL       ;DO A READ HEADER
021712  012762  000025  000000                 MOV     #25,RKCS1(R2)   ;DO A READ HDR, WITH FMT BIT = 0
021720  032762  000200  000000         24$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
021726  001774                               BEQ     24$
021730  012762  010025  000000                 MOV     #10025,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
021736  032762  000200  000000         26$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
021744  001774                               BEQ     26$
021746  105037  003124                         CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
```

```
 4970  021752  005762  000024                 TST     RKDB(R2)          ;POP SILO ONE TIME
 4971  021756  032762  001000  000024         BIT     #BIT9,RKDB(R2)    ;TEST FORMAT BIT IN HEADER WORD 2
 4972  021764  001403                         BEQ     28$               ;BR IF 22 SECTOR FORMAT
 4973  021766  152737  000020  003124         BISB    #8.CFMT,FORMAT    ;SET 20 SECTOR FORMAT FOR THIS DRIVE
 4974  021774  004737  020616          28$:   JSR     PC,INITSS         ;INIT THE S.S.
 4975  022000  004737  025676                 JSR     PC,REDBSF         ;READ BAD SECTOR FILE FOR THIS DRIVE
 4976  022004  105737  003116                 TSTB    MDFLAG            ;SEE IF 200 START
 4977  022010  001004                         BNE     32$               ;BR IF NOT 200 START
 4978  022012  122737  000061  003145         CMPB    #'1,PASSNO        ;SEE IF FIRST PASS
 4979  022020  001012                         BNE     30$               ;BR IF NOT FIRST PASS
 4980  022022  004737  023320          32$:   JSR     PC,DRVSER         ;TYPE "DRIVE SER. NO.  XXX"
 4981  022026  004737  023440                 JSR     PC,CRTSER         ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
 4982  022032  032777  000200  157100         BIT     #BIT7,@SWR        ;SEE IF BAD SECTORS SHOULD BE TYPED
 4983  022040  001402                         BEQ     30$               ;BR IF NOT
 4984  022042  004737  026164                 JSR     PC,TYPBSF         ;TYPE BAD SECTOR FILES
 4985  022046  005037  005432          30$:   CLR     INTCHR            ;INIT. TTY INPUT CHAR BUFFER
 4986  022052  000207                         RTS     PC                ;RETURN
 4987
 4988
 4989
 4990                                  ;;*****************************************************************
 4991                                  .SBTTL  DMCART - DISMOUNT TEST CARTRIDGE
 4992                                  ;*THIS SUBROUTINE DIRECTS THE OPERATOR IN THE DISMOUNTING
 4993                                  ;*OF THE TEST CARTRIDGE ON THE DRIVE CURRENTLY UNDER TEST.
 4994                                  ;;*****************************************************************
 4995  022054
 4996  022054  013700  005544          DMCART: MOV     DRVPTR,R0         ;GET DRIVE LIST POINTER
 4997  022060  112037  010173                  MOVB    (R0)+,DISPAK+15.  ;PUT DRIVE NAME INTO MSG
 4998  022064  111037  010172                  MOVB    (R0),DISPAK+14.
 4999  022070  004737  020616                  JSR     PC,INITSS         ;INITIALIZE SUBSYS
 5000  022074  042762  000100  000000         BIC     #IE,RKCS1(R2)     ;DISABLE RK06 INTRPT FOR MANUAL INTERVENTION
 5001  022102  104401  010154                  TYPE    ,DISPAK           ;TYPE "UNLOAD DRIVE XX AND REMOVE PACK"
 5002  022106  104401  010352          5$:     TYPE    ,TYPRWN           ;TYPE "TYPE R<CR> WHEN "
 5003  022112  104401  010412                  TYPE    ,DRIDON           ;TYPE "DONE :"
 5004  022116  004737  023006                  JSR     PC,RDCHRS         ;WAIT FOR R<CR> RESPONSE
 5005  022122  022146                          12$                       ;(↑C) RETURN ADDRESS
 5006  022124  022170                          16$                       ;(↑Z) RETURN ADDRESS
 5007  022126  022106                          5$                        ;(↑U) RETURN ADDRESS
 5008  022130  005700                          TST     R0                ;SEE IF NULL INPUT
 5009  022132  001025                          BNE     8$                ;BR IF NOT NULL
 5010  022134  104401  005202          6$:     TYPE    ,BUFFO            ;ECHO BAD INPUT
 5011  022140  104401  001324                  TYPE    ,$QUES            ;TYPE (?) AND (CR),(LF)
 5012  022144  000760                          BR      5$                ;GO ASK AGAIN
 5013  022146  012706  001100          12$:    MOV     #STACK,SP         ;RE-INIT THE STACK
 5014  022152  105737  003116                  TSTB    MDFLAG            ;SEE IF 200 START
 5015  022156  001002                          BNE     14$               ;BR IF NOT
 5016  022160  000137  011746                  JMP     DFSTRT            ;RESTART PROGRAM - 200 START
 5017  022164  000137  012546          14$:    JMP     ALLSYS            ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
 5018  022170  105737  003116          16$:    TSTB    MDFLAG            ;SEE IF 200 START
 5019  022174  001744                          BEQ     5$                ;BR IF YES, TO ASK AGAIN
 5020  022176  012706  001100                  MOV     #STACK,SP         ;RE-INIT THE STACK
 5021  022202  000137  013212                  JMP     ASKSYS            ;GO ASK FOR SUBSYS TO TEST AGAIN
 5022  022206  023727  005202  000122  8$:     CMP     BUFFO,#'R         ;SEE IF R<CR> TYPED
 5023  022214  001347                          BNE     6$                ;BR IF NOT
 5024  022216  004737  020616                  JSR     PC,INITSS         ;INITIALIZE S.S.
 5025  022222  000207                          RTS     PC                ;RETURN
```

```
                                ;*****************************************************************
                                ;SBTTL  INTBLD - INITIALIZE TABLED
                                ;*THIS SUBROUTINE LOADS TABLED WITH THE INITIAL PATTERN OF DRIVE NUMBERS
                                ;*WHICH IS THE CYLINDER BLOCK LAYOUT FOR THE FIRST CURRENT ZONE. IT USES
                                ;*THE STARTING DRIVE NUMBER LIST, STDLST, FOR THE FIRST COLUMN, AND
                                ;*INCREMENTS THESE DRIVE NUMBERS ACROSS THE TABLE, FORMING ROWS. FOR
                                ;*ANY DRIVE WHICH IS NOT PRESENT, THE ENTRY IS SET = 377.
                                ;*****************************************************************
022224  104407                  INTBLD: SAVREG                  ;SAVE R0-R5
                                ;LOAD THE FIRST COLUMN OF TABLED FROM STDLST
022226  012700  006410                  MOV     #STDLST,R0      ;STARTING ADRS OF STDLST
022232  012701  005750                  MOV     #TABLED,R1      ;STARTING ADRS OF TABLED
022236  012703  000020                  MOV     #16.,R3         ;SET COUNTER = 16
022242  112011                  4$:     MOVB    (R0)+,(R1)      ;LOAD A BYTE INTO TABLED
022244  062701  000020                  ADD     #16.,R1         ;INCREMENT TABLED POINTER
022250  005303                          DEC     R3              ;DECR COUNTER
022252  001373                          BNE     4$              ;BR IF NOT DONE YET
                                ;GET INDEX OF FIRST ZERO ENTRY IN DRVLST INTO R1
022254  012700  005502                  MOV     #DRVLST,R0      ;STARTING ADRS OF DRVLST
022260  005001                          CLR     R1              ;INIT INDEX TO 0
022262  005720                  6$:     TST     (R0)+           ;SEE IF ENTRY = 0
022264  001404                          BEQ     8$              ;BR IF YES
022266  005201                          INC     R1              ;INCREMENT POINTER
022270  020127  000020                  CMP     R1,#16.         ;SEE IF DONE LOOKING YET
022274  002772                          BLT     6$              ;BR IF NOT DONE YET
                                ;INCREMENT THE DRIVE NOS. TO FORM THE ROWS
022276  012700  005750          8$:     MOV     #TABLED,R0      ;INIT TABLED POINTER
022302  012704  000017          16$:    MOV     #15.,R4         ;INIT ROW ELEMENT COUNTER
022306  112003                          MOVB    (R0)+,R3        ;INIT ROW ELEMENT VALUE
022310  005203                  18$:    INC     R3              ;INCR ROW ELEMENT VALUE
022312  020327  000020                  CMP     R3,#20          ;SEE IF ROW ELEMENT IS VALID
022316  002401                          BLT     20$             ;BR IF VALUE IS OK
022320  005003                          CLR     R3              ;RESET VALUE TO 0
022322  020301                  20$:    CMP     R3,R1           ;SEE IF VALUE IS > ACTUAL DRIVES
022324  002403                          BLT     22$             ;BR IF VALUE OK
022326  112720  000377                  MOVB    #377,(R0)+      ;SET INVALID ENTRY = 377
022332  000401                          BR      24$             ;PROCEED
022334  110320                  22$:    MOVB    R3,(R0)+        ;SET VALUE IN TABLE
022336  005304                  24$:    DEC     R4              ;DECR ROW ELEMENT COUNTER
022340  001363                          BNE     18$             ;BR IF NOT DONE WITH ROW YET
022342  020027  006350                  CMP     R0,#TABLED+256. ;SEE IF DONE WITH TABLE YET
022346  103755                          BLO     16$             ;BR IF NOT DONE YET
                                ;SET INVALID ELEMENTS OF FIRST COLUMN = 377
022350  012700  005750                  MOV     #TABLED,R0      ;INIT COLUMN POINTER
022354  012703  000020                  MOV     #16.,R3         ;INIT COL. ELEMENT COUNTER
022360  121001                  26$:    CMPB    (R0),R1         ;SEE IF ELEMENT IS VALID
022362  002402                          BLT     30$             ;BR IF YES
022364  112710  000377                  MOVB    #377,(R0)       ;SET INVALID ELEMENT = 377
022370  062700  000020          30$:    ADD     #16.,R0         ;INCREMENT COLUMN POINTER
022374  005303                          DEC     R3              ;SEE IF ALL DONE YET
022376  001370                          BNE     26$             ;BR IF NOT ALL DONE YET
022400  104410                          RESREG                  ;RESTORE R0-R5
022402  000207                          RTS     PC              ;RETURN
```

```
5082
5083
5084
5085                                    ;;*****************************************************************
5086                                    .SBTTL  ROTBLD - ROTATE TABLED RIGHT 3 COLUMNS
5087                                    ;*THIS SUBROUTINE ROTATES ALL ELEMENTS OF TABLED RIGHT 3 COLUMNS,
5088                                    ;*SO THAT THE DRIVE DATA WILL BE ROTATED SEVERAL SECTORS IN EACH
5089                                    ;*CURRENT ZONE.
5090                                    ;;*****************************************************************
5091  022404  104407                    ROTBLD: SAVREG                          ;SAVE R0-R5
5092  022406  012700  006350                    MOV     #TABLED+256.,R0         ;POINT TO END OF TABLED
5093  022412  010001                             MOV     R0,R1                   ;GET A COPY IN R1
5094  022414  012704  000015            4$:     MOV     #13.,R4                 ;INIT ROW ELEMENT COUNTER
5095  022420  114037  001304                     MOVB    -(R0),$TMP11            ;SAVE LAST 3 ELEMENTS IN THIS ROW
5096  022424  114037  001306                     MOVB    -(R0),$TMP12
5097  022430  114037  001310                     MOVB    -(R0),$TMP13
5098  022434  114041                    6$:     MOVB    -(R0),-(R1)             ;MOVE AN ELEMENT RIGHT 3 PLACES
5099  022436  005304                             DEC     R4                      ;DECR ROW ELEMENT COUNTER
5100  022440  001375                             BNE     6$                      ;BR IF NOT DONE YET
5101  022442  113741  001304                     MOVB    $TMP11,-(R1)            ;MOVE IN 3 SAVED ELEMENTS
5102  022446  113741  001306                     MOVB    $TMP12,-(R1)
5103  022452  113741  001310                     MOVB    $TMP13,-(R1)
5104  022456  020027  005750                     CMP     R0,#TABLED              ;SEE IF DONE WITH TABLE YET
5105  022462  101354                             BHI     4$                      ;BR IF NOT DONE YET
5106  022464  104410                             RESREG                          ;RESTORE R0-R5
5107  022466  000207                             RTS     PC                      ;RETURN
5108
5109
5110
5111                                    ;;*****************************************************************
5112                                    .SBTTL  WRTBLK - WRITE ONE CYLINDER BLOCK
5113                                    ;*THIS SUBROUTINE WRITES ALL THE SECTORS FOR THE CURRENT DRIVE
5114                                    ;*INTO THE CYLINDER BLOCK WHOSE STARTING CYL NUMBER IS IN
5115                                    ;*P.CYLN(R5) ON ENTRY. THE DATA IS WRITTEN ON ALL  SURFACES
5116                                    ;*THE LOGICAL DRIVE NUMBER MUST BE IN LOGDRV ON ENTRY.
5117                                    ;*
5118                                    ;*IT CALLS WRTSEC TO ACTUALLY WRITE EACH 256-WORD SECTOR.
5119                                    ;;*****************************************************************
5120  022470  104407                    WRTBLK: SAVREG                          ;SAVE R0-R5
5121  022472  113700  003144                    MOVB    LOGDRV,R0               ;GET LOGICAL DRIVE NO.
5122  022476  006300                             ASL     R0                      ;CONVERT IT TO DATA PATTERN INDEX
5123  022500  016003  006350                     MOV     TABLEG(R0),R3           ;GET DATA PATTERN WORD INTO R3
5124  022504  012701  005750                     MOV     #TABLED,R1              ;GET TABLED POINTER
5125  022510  012704  000020                     MOV     #16.,R4                 ;INIT CYLINDER COUNTER
5126  022514  010100                    4$:     MOV     R1,R0
5127  022516  123720  003144            6$:     CMPB    LOGDRV,(R0)+            ;SEE IF DRIVE LISTED HERE
5128  022522  001375                             BNE     6$                      ;BR IF NOT FOUND YET IN THIS ROW
5129  022524  005300                             DEC     R0                      ;BACK UP  BY 1 BYTE
5130  022526  160100                             SUB     R1,R0                   ;COMPUTE ROW POSITION
5131  022530  116065  006430  000004             MOVB    SECLST(R0),P.SECT(R5) ;GET SECTOR NO. FROM ROW POSITION
5132  022536  105065  000005                     CLRB    P.TRCK(R5)              ;SET TRACK = 0
5133  022542  004737  025126            8$:     JSR     PC,WRTSEC               ;WRITE 256 WORDS AT THIS SECTOR
5134  022546  105265  000005                     INCB    P.TRCK(R5)              ;INCR TRACK NO.
5135  022552  126527  000005  000002             CMPB    P.TRCK(R5),#2           ;SEE IF ALL TRACKS DONE YET
5136  022560  003770                             BLE     8$                      ;BR IF NOT DONE YET
5137  022562  062701  000020                     ADD     #16.,R1                 ;POINT TO START OF NEXT ROW
```

```
5138  022566  005265  000002                  INC     P.CYLN(R5)      ;INCR CYLINDER NO.
5139  022572  005304                          DEC     R4              ;DECREMENT CYLINDER COUNTER
5140  022574  001347                          BNE     4$              ;BR IF 16 CYLS NOT DONE YET
5141  022576  104410                          RESREG                  ;RESTORE R0-R5
5142  022600  000207                          RTS     PC              ;RETURN
5143
5144
5145
5146                                  ;;*****************************************************************
5147                                  ;#CKSCOR - DECREMENT DRIVE SCORES ROUTINE
5148                                  ;#THIS SUBROUTINE CHECKS TO SEE IF THE PROGRAM IS CURRENTLY
5149                                  ;#READING OVRWRT OR COMPAT DATA WITH OFFSETS AND HANDLING DATA CHECK
5150                                  ;#ERRORS. IF SO, THE SCORE FOR THE DRIVE WHICH WROTE THE
5151                                  ;#FAILING DATA IS DECREMENTED.
5152                                  ;#ON ENTRY, THE BYTE LABELED "TSTTYP" HAS THE FOLLOWING
5153                                  ;#POSSIBLE VALUES :
5154                                  ;#       TSTTYP = 0  - NOT PRESENTLY READING WITH OFFSET
5155                                  ;#              = 1  - PRESENTLY READING OVRWRT DATA WITH OFFSET
5156                                  ;#              = 2  - PRESENTLY READING COMPAT DATA WITH OFFSET
5157                                  ;#              = 3  - PRESENTLY READING SELF TEST DATA WITH OFFSET
5158                                  ;;*****************************************************************
5159  022602  104407                  CKSCOR: SAVREG                  ;SAVE R0-R5
5160  022604  105737  003133                  TSTB    TSTTYP          ;SEE IF CURRENTLY TESTING WITH OFFSET
5161  022610  001474                          BEQ     20$             ;BR TO EXIT, IF NOT
5162  022612  032737  000024  005414          BIT     #DCKERR!HVRCER,RECODE ;SEE IF A DCK OR HVRC ERR OCCURRED
5163  022620  001470                          BEQ     20$             ;BR IF NOT
5164                                  ;GET THE NO. OF THE DRIVE WHICH WROTE DATA WHICH CAUSED DCK OR HVRC ERR
5165  022622  004737  033444                  JSR     PC,GTPKAD       ;COMPUTE PACK ADDRESS OF ERROR
5166  022626  122737  000003  003133          CMPB    #3,TSTTYP       ;SEE IF DOING SELF-TEST
5167  022634  001002                          BNE     4$              ;BR IF NOT
5168  022636  005004                          CLR     R4              ;SET DRIVE NO. = 0
5169  022640  000431                          BR      10$             ;CONTINUE
5170  022642  013701  001174          4$:     MOV     $REG5,R1        ;GET CYL NO. OF ERROR
5171  022646  042701  177760                  BIC     #177760,R1      ;CLEAR HI CYL BITS
5172  022652  006301                          ASL     R1              ;CONVERT TO TABLED INDEX FOR ROW
5173  022654  006301                          ASL     R1
5174  022656  006301                          ASL     R1
5175  022660  006301                          ASL     R1
5176  022662  062701  005750                  ADD     #TABLED,R1      ;COMPUTE POINTER TO TABLED ROW
5177  022666  005000                          CLR     R0              ;INIT SECLST INDEX TO 0
5178  022670  123760  001200  006430  6$:     CMPB    $REG7,SECLST(R0) ;SEE IF SECTOR IS LISTED IN SECLST
5179  022676  001405                          BEQ     8$              ;BR IF YES
5180  022700  005200                          INC     R0              ;INCR SECLST INDEX
5181  022702  020027  000020                  CMP     R0,#16.         ;SEE IF WHOLE LIST CHECKED YET
5182  022706  002770                          BLT     6$              ;BR IF NOT YET
5183  022710  000434                          BR      20$             ;SECTOR NOT LISTED - EXIT, BECAUSE
5184                                                                  ;  FAILING DRIVE IS NOT BEING TESTED
5185  022712  060001                  8$:     ADD     R0,R1           ;COMPUTE TABLED POINTER TO FAILING DRIVE
5186  022714  111104                          MOVB    (R1),R4         ;GET ACTUAL DRIVE NO.
5187  022716  122704  000377                  CMPB    #377,R4         ;SEE IF DRIVE IS NOT BEING TESTED
5188  022722  001427                          BEQ     20$             ;BR IF DRIVE NOT TESTED
5189  022724  012700  007270          10$:    MOV     #NSCORO,R0      ;GET ADRS OF NEG SCRATCH SCORES
5190  022730  105737  003146                  TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
5191  022734  001402                          BEQ     12$             ;BR IF NEG
5192  022736  012700  007350                  MOV     #PSCORO,R0      ;GET ADRS OF POS SCRATCH SCORES
5193  022742  105737  001176          12$:    TSTB    $REG6           ;SEE IF TRACK = 0
```

```
                                        BEQ     14$             ;BR IF YES
                                        ADD     #16.,R0         ;POINT TO SCORES FOR TRACK 1
                                        CMPB    $REG6,#1        ;SEE IF TRACK = 1
                                        BEQ     14$             ;BR IF YES
                                        ADD     #16.,R0         ;POINT TO SCORES FOR TRACK 2
                                14$:    ADD     R4,R0           ;COMPUTE POINTER TO SCORE FOR DRIVE
                                        CMPB    (R0),OFSCNT     ;SEE IF THIS SCORE IS PERFECT SO FAR
                                        BNE     20$             ;EXIT IF NOT
                                        DECB    (R0)            ;DECREMENT SCORE FOR THIS DRIVE
                                20$:    RESREG                  ;RESTORE R0-R5
                                        RTS     PC              ;RETURN


;;*****************************************************************
;SBTTL  RDCHRS - READ A STRING OF KBD INPUT CHARS
;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
;*THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
;*RUB-OUT AND (↑U) FEATURES ARE PROVIDED.
;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
;*TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
;*TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
;*IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
;*FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
;*IS TAKEN.
;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
;*IN R0.
;*
;*  CALL -      JSR     PC,RDCHRS
;*              <CONTROL-C RETURN ADDRESS>
;*              <CONTROL-Z RETURN ADDRESS>
;*              <CONTROL-U OR ERROR RETURN ADDRESS>
;*              RETURN
;;*****************************************************************

RDCHRS:
                        MOV     R1,-(SP)        ;SAVE R1
                        MOV     R2,-(SP)        ;SAVE R2
                        CLR     R0              ;INITIALIZE CHARACTER COUNT
                        CLR     R1              ;INITIALIZE RUB-OUT INDICATOR
;READ A CHARACTER
2$:                     RDCHR                   ;READ A CHARACTER
                        MOVB    (SP)+,R2        ;GET CHARACTER INTO R2
;CHECK FOR (↑C)
                        CMPB    #003,R2         ;SEE IF (↑C) TYPED
                        BNE     4$              ;BR IF NOT (↑C)
                        TYPE    ,CNTRLC         ;ECHO (↑C)
                3$:     MOV     @4(SP),4(SP)    ;PUT RETURN ADDRESS ON STACK
                        BR      24$             ;BR TO TAKE EXIT
;CHECK FOR (↑Z)
                4$:     CMPB    #032,R2         ;SEE IF (↑Z) TYPED
                        BNE     6$              ;BR IF NOT (↑Z)
                        TYPE    ,CNTRLZ         ;ECHO (↑Z)
                        ADD     #2,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADR.
```

```
5250  023064  000763                          BR      3$              ;BR TO TAKE (↑Z) EXIT
5251                                  ;CHECK FOR (↑U)
5252  023066  122702  000025          6$:     CMPB    #025,R2         ;SEE IF (↑U) TYPED
5253  023072  001006                          BNE     8$              ;BR IF NOT (↑U)
5254  023074  104401  011671                  TYPE    ,CNTRLU         ;ECHO (↑U)
5255  023100  062766  000004  000004  7$:     ADD     #4,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
5256  023106  000752                          BR      3$              ;BR TO TAKE (↑U) EXIT
5257                                  ;CHECK FOR (↑G)
5258  023110  122702  000007          8$:     CMPB    #007,R2         ;SEE IF (↑G) TYPED
5259  023114  001005                          BNE     9$              ;BR IF NOT (↑G)
5260  023116  104401  011676                  TYPE    ,CNTRLG         ;ECHO (↑G)
5261  023122  004737  020564                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
5262  023126  000764                          BR      7$              ;TAKE (↑U) RETURN
5263                                  ;CHECK FOR RUB-OUT (DELETE)
5264  023130  122702  000177          9$:     CMPB    #177,R2         ;SEE IF RUB-OUT (DEL) TYPED
5265  023134  001020                          BNE     14$             ;BR IF NOT RUB-OUT
5266  023136  005700                          TST     R0              ;CHECK THE CHARACTER COUNT
5267  023140  001726                          BEQ     2$              ;BR IF COUNT = 0
5268  023142  005701                          TST     R1              ;CHECK THE RUB-OUT INDICATOR
5269  023144  001003                          BNE     11$             ;BR IF WE HAD A PREVIOUS RUB-OUT
5270  023146  005201                          INC     R1              ;SET RUB-OUT INDICATOR
5271  023150  104401  011707                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH (\)
5272  023154  005037  005434          11$:    CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
5273  023160  005300                          DEC     R0              ;DECREMENT COUNT
5274  023162  116037  005202  005434          MOVB    BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
5275  023170  104401  005434                  TYPE    ,SCRACH         ;ECHO CHARACTER TO BE DELETED
5276  023174  000710                          BR      2$              ;GO READ ANOTHER CHARACTER
5277  023176  005701                  14$:    TST     R1              ;CHECK THE RUB-OUT INDICATOR
5278  023200  001403                          BEQ     16$             ;BR IF INDICATOR IS NOT SET
5279  023202  104401  011707                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH
5280  023206  005001                          CLR     R1              ;CLEAR THE RUB-OUT INDICATOR
5281                                  ;CHECK FOR CARRIAGE RETURN
5282  023210  122702  000015          16$:    CMPB    #015,R2         ;SEE IF <CR> TYPED
5283  023214  001426                          BEQ     19$             ;BR IF <CR>
5284                                  ;HANDLE POSSIBLE DIGIT
5285  023216  005037  005434                  CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
5286  023222  110237  005434                  MOVB    R2,SCRACH       ;GET THIS CHARACTER INTO BUFFER
5287  023226  104401  005434                  TYPE    ,SCRACH         ;ECHO THE CHARACTER TYPED
5288  023232  110260  005202                  MOVB    R2,BUFFO(R0)    ;PUT CHARACTER INTO BUFFER
5289  023236  005200                          INC     R0              ;INCREMENT CHARACTER COUNTER
5290  023240  022700  000120                  CMP     #80.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
5291  023244  001264                          BNE     2$              ;BR IF NOT TOO MANY
5292  023246  104401  001325                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
5293  023252  112760  000000  005202          MOVB    #0,BUFFO(R0)    ;PUT TERMINATING NULL INTO BUFFER
5294  023260  104401  005202                  TYPE    ,BUFFO          ;ECHO INPUT STRING
5295  023264  104401  001324                  TYPE    ,$QUES          ;TYPE (?) <CR> <LF>
5296  023270  000703                          BR      7$              ;TAKE ERROR EXIT FROM RDCHRS
5297  023272  104401  001325          19$:    TYPE    ,$CRLF          ;TYPE <CR> <LF>
5298  023276  112760  000000  005202          MOVB    #0,BUFFO(R0)    ;PUT TERMINATING NULL INTO BUFFER
5299  023304  062766  000006  000004          ADD     #6,4(SP)        ;FIX UP RETURN ADDRESS
5300  023312  012602                  24$:    MOV     (SP)+,R2        ;RESTORE R2
5301  023314  012601                          MOV     (SP)+,R1        ;RESTORE R1
5302  023316  000207                          RTS     PC              ;SUBROUTINE EXIT
5303
5304
5305
```

```
                                ;;*****************************************************************
                                .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
                                ;*THIS SUBROUTINE TYPES "DRIVE SER. NO.  XXX" (IN DECIMAL), WITH LEADING
                                ;*ZEROS SUPPRESSED.
                                ;;*****************************************************************
023320  104407                  DRVSER: SAVREG                  ;SAVE R0-R5
023322  004737  020616                  JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
023326  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
023334  004737  027302                  JSR     PC,DRVCAL       ;READ STATUS OF THIS DRIVE
023340  104401  011501                  TYPE    ,DRIV           ;TYPE "DRIVE"
023344  104401  011517                  TYPE    ,SERNM          ;TYPE " SER. NO.  "
023350  016501  000054                  MOV     P.AII(R5),R1    ;GET "A" STATUS BYTE 11
023354  012704  042312                  MOV     #$OCTVL,R4      ;GET ADDR OF CHAR BUFFER
023360  010446                          MOV     R4,-(SP)        ;STORE IT ON STACK FOR $SUPRS
023362  012703  000003                  MOV     #3,R3           ;INIT CHAR COUNT
023366  006101                          ROL     R1              ;INITIALIZE BIT POSITIONS
023370  006101                          ROL     R1
023372  006101                  4$:     ROL     R1              ;GET NEXT 4 BITS
023374  006101                          ROL     R1
023376  006101                          ROL     R1
023400  006101                          ROL     R1
023402  010100                          MOV     R1,R0           ;GET A WORKING COPY
023404  042700  177760                  BIC     #177760,R0      ;CLEAR ALL BUT LOW 4 BITS
023410  052700  000060                  BIS     #'0,R0          ;CONVERT A DIGIT TO ASCII
023414  110024                          MOVB    R0,(R4)+        ;PUT ASCII DIGIT INTO CHAR BUFFER
023416  005303                          DEC     R3              ;DECREMENT CHAR COUNT
023420  001364                          BNE     4$              ;BR IF NOT 3 CHARS YET
023422  105014                          CLRB    (R4)            ;INSERT NULL TERMINATOR
023424  004737  042524                  JSR     PC,@#$SUPRS     ;TYPE DRIVE SER. NUMBER
023430  104401  001325                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
023434  104410                          RESREG                  ;RESTORE R0-R5
023436  000207                          RTS     PC              ;RETURN
023440
023442
023444
                                ;;*****************************************************************
                                .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
                                ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXXX" (IN OCTAL),
                                ;*WITH LEADING ZEROS SUPPRESSED.
                                ;;*****************************************************************
023446  004737  020616          CRTSER: JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
023452  105065  000007                  CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
023456  105737  003124                  TSTB    FORMAT          ;CHECK THE ACTUAL FORMAT
023462  001402                          BEQ     4$              ;BR IF 22 SECTORS
023464  105265  000004                  INCB    P.SECT(R5)      ;IF 20 SECTORS, READ SECTOR 1
023470  112765  000121  000001  4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
023476  012765  000632  000002          MOV     #LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)
023504  112765  000002  000005          MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
023512  012765  052476  000010          MOV     #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
023520  012765  177776  000012          MOV     #-2,P.WC(R5)    ;SET WORD COUNT TO READ 2 WORDS
023526  004737  027302                  JSR     PC,DRVCAL       ;READ SERIAL NO. IN BSF
023532  104401  011510                  TYPE    ,CART           ;TYPE "CART."
023536  104401  011517                  TYPE    ,SERNM          ;TYPE "SER. NO.  "
023542  012746  052476                  MOV     #RWBUF,-(SP)    ;GET POINTER FOR $DB2O
023546  004737  042210                  JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL
023552  004737  042524                  JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
```

```
5362  023550  104401  001325                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
5363  023554  000207                          RTS     PC              ;RETURN
5364
5365
5366
5367
5368
5369
5370                          ;;*****************************************************************
5371                          .SBTTL  STALL - STALL FOR ST UNIT STALL TIMES
5372                          ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
5373                          ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
5374                          ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
5375                          ;* CALL -   JSR     PC,STALL
5376                          ;;*****************************************************************
5377
5378  023556  010046                  STALL:  MOV     R0,-(SP)        ;SAVE R0
5379  023560  010146                          MOV     R1,-(SP)        ;SAVE R1
5380  023562  032777  000400  155350          BIT     #BIT08,@SWR     ;APPLY RANDOM STALL ?
5381  023570  001407                          BEQ     1$              ;BR IF NOT RANDOM
5382  023572  004737  041406                  JSR     PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
5383  023576  013700  041506                  MOV     $LONUM,R0       ;GET IT INTO R0
5384  023602  006200                          ASR     R0              ;SCALE IT DOWN
5385  023604  006200                          ASR     R0
5386  023606  000403                          BR      2$              ;GO STALL WITH RANDOM NO.
5387  023610  013700  005422          1$:     MOV     STALLS,R0       ;GET REQUESTED NO. OF STALLS
5388  023614  001406                          BEQ     6$              ;RETURN IF NO STALL REQUIRED
5389  023616  012701  000016          2$:     MOV     #14.,R1         ;SET CONSTANT FOR 40 US
5390  023622  005301                  4$:     DEC     R1              ;INNER LOOP COUNTER
5391  023624  001376                          BNE     4$              ;INNER LOOP BR
5392  023626  005300                          DEC     R0              ;OUTER LOOP COUNTER
5393  023630  001372                          BNE     2$              ;OUTER LOOP BR
5394  023632  012601                  6$:     MOV     (SP)+,R1        ;RESTORE R1
5395  023634  012600                          MOV     (SP)+,R0        ;RESTORE R0
5396  023636  000207                          RTS     PC              ;RETURN
5397
5398
5399
5400                          ;;*****************************************************************
5401                          ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF THE DATA PATTERN
5402                          ;*IN TABLEB INTO ALL 256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
5403                          ;;*****************************************************************
5404  023640  104407          LODSEC: SAVREG                          ;SAVE R0-R5
5405  023642  012701  052476          MOV     #RWBUF,R1               ;GET ADRS OF R/W BUF INTO R1
5406  023646  012700  005656  2$:     MOV     #TABLEB,R0              ;GET DATA PATTERN ADDRESS
5407  023652  012703  000020          MOV     #16.,R3                 ;INIT PATTERN COUNTER
5408  023656  012021          4$:     MOV     (R0)+,(R1)+             ;MOVE A PATTERN WORD
5409  023660  020127  053476          CMP     R1,#RWBUF+512.          ;SEE IF DONE YET
5410  023664  103003                  BHIS    9$                      ;BR IF DONE
5411  023666  005303                  DEC     R3                      ;DECREMENT PATTERN COUNTER
5412  023670  001372                  BNE     4$                      ;BR IF NOT RESTARTING PATTERN YET
5413  023672  000765                  BR      2$                      ;BR TO RESTART PATTERN
5414  023674  104410          9$:     RESREG                          ;RESTORE R0-R5
5415  023676  000207                  RTS     PC                      ;RETURN
5416
5417
```

```
                                ;;*****************************************************************
                                .SBTTL  FINADR - COMPUTE FINAL PACK ADDRESS
                                ;*THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
                                ;*COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
                                ;*WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
                                ;*P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
                                ;*PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
                                ;*THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
                                ;*DATA MISCOMPARE.
                                ;;*****************************************************************
023700  104407                  FINADR: SAVREG                          ;SAVE R0-R5
023702  013746  005464                  MOV     LASTWC,-(SP)            ;STORE WORD COUNT
023706  005416                          NEG     (SP)                    ;MAKE IT POSITIVE
023710  005046                  18$:    CLR     -(SP)                   ;MAKE ROOM ON STACK
023712  116616  000003                  MOVB    3(SP),(SP)              ;STORE NO. OF SECTORS TRANSFERRED
023716  005066  000002                  CLR     2(SP)                   ;CLEAR LOCATION ON STACK
023722  116566  000004  000002          MOVB    P.SECT(R5),2(SP)        ;STORE STARTING SECTOR
023730  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL SECTOR ADDRESS
023734  005066  000002                  CLR     2(SP)                   ;CLEAR NO. OF TRACKS TRANSFERRED
023740  012700  000026                  MOV     #22.,R0                 ;SET FOR 22 SECTORS
023744  105737  003124                  TSTB    FORMAT                  ;DETERMINE THE FORMAT
023750  001402                          BEQ     19$                     ;BR IF 22 SECTORS
023752  012700  000024                  MOV     #20.,R0                 ;SET FOR 20 SECTORS
023756  020016                  19$:    CMP     R0,(SP)                 ;CHECK FOR SECTOR OVERFLOW
023760  101004                          BHI     20$                     ;NO, CHECK IF SECTOR CORRECT
023762  160016                          SUB     R0,(SP)                 ;DECREMENT SECTOR COUNT BY 20 OR 22
023764  005266  000002                  INC     2(SP)                   ;INCREMENT TRACKS TRANSFERRED
023770  000772                          BR      19$                     ;CHECK FOR SECTOR OVERFLOW
023772  112637  005463          20$:    MOVB    (SP)+,FINSEC            ;STORE FINAL SECTOR
023776  005046                          CLR     -(SP)                   ;MAKE ROOM FOR TRACKS TRANSFERRED
024000  116516  000005                  MOVB    P.TRCK(R5),(SP)         ;STORE STARTING TRACK
024004  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL TRACK ADDRESS
024010  005066  000002                  CLR     2(SP)                   ;CLEAR FINAL CYLINDER
024014  122716  000003          21$:    CMPB    #3,(SP)                 ;CHECK FOR TRACK OVERFLOW
024020  101005                          BHI     22$                     ;NO, CHECK FINAL TRACK
024022  162716  000003                  SUB     #3,(SP)                 ;DECREMENT TRACK COUNT BY 3
024026  005266  000002                  INC     2(SP)                   ;INCR CYL COUNT
024032  000770                          BR      21$                     ;CHECK FOR TRACK OVERFLOW
024034  112637  005462          22$:    MOVB    (SP)+,FINTRK            ;STORE FINAL TRACK
024040  066516  000002                  ADD     P.CYLN(R5),(SP)         ;CALCULATE FINAL CYLINDER
024044  011637  005460                  MOV     (SP),FINCYL             ;STORE FINAL  CYLINDER
024050  005726                          TST     (SP)+                   ;CLEAN OFF STACK
024052  104410                          RESREG                          ;RESTORE R0-R5
024054  000207                          RTS     PC                      ;RETURN


                                ;;*****************************************************************
                                .SBTTL  LODBUF - LOAD THE READ/WRITE DATA BUFFER
                                ;*THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
                                ;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
                                ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
```

```
                                ;*OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
                                ;*IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
                                ;*OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
                                ;*ENTRY, ARE LOADED INTO THE BUFFER.
                                ;;*****************************************************************

024056  104407                  LODBUF: SAVREG                  ;SAVE R0-R5
024060  013702  005456                  MOV     WDSXFR,R2       ;GET NO. OF WORDS
024064  005737  005436                  TST     PATRN           ;SEE IF QUICK VERIFY DATA TEST DESIRED
024070  001007                          BNE     3$              ;BR IF NOT QUICK VERIFY
024072  012700  005556                  MOV     #PAT00,R0       ;SET DATA PATTERN STARTING ADDRESS
024076  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
024102  012701  052476                  MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
024106  000422                          BR      30$             ;PROCEED
024110  005000                  3$:     CLR     R0              ;INIT PATTERN NUMBER
024112  032701  000001          4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
024116  001003                          BNE     6$              ;BR IF THIS BIT IS SET
024120  005200                          INC     R0              ;INCREMENT PATTERN NO.
024122  006201                          ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
024124  000772                          BR      4$              ;BR TO CHECK NEXT BIT
024126  006300                  6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
024130  006300                          ASL     R0
024132  006300                          ASL     R0
024134  006300                          ASL     R0
024136  006300                          ASL     R0
024140  062700  005556                  ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
024144  012746  000020                  MOV     #16.,-(SP)      ;SET PATTERN WORD COUNT
024150  013701  005474                  MOV     PMA,R1          ;SET BUFFER ADDRESS
024154  010003                  30$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
024156  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
024160  012321                  34$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
024162  005302                          DEC     R2              ;DECREMENT WORD COUNTER
024164  001403                          BEQ     44$             ;BR IF ALL DONE
024166  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
024170  001373                          BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
024172  000770                          BR      30$             ;BR TO REPEAT THE PATTERN
024174  005726                  44$:    TST     (SP)+           ;POP THE STACK
024176  104410                          RESREG                  ;RESTORE R0-R5
024200  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  CMPBUF - SOFTWARE COMPARE DATA
                                ;*THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
                                ;*TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
                                ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
                                ;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
                                ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
                                ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
                                ;;*****************************************************************
024202  104407                  CMPBUF: SAVREG                  ;SAVE R0-R5
024204  112737  000040  051021          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
024212  012737  000007  052370          MOV     #7,DF25+2
024220  013702  005456                  MOV     WDSXFR,R2       ;SET NO. OF WORDS
024224  005037  005434                  CLR     SCRACH          ;CLEAR COMPARE ERROR COUNT
```

```
5530  024230  004737  030360                JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
5531  024234  005737  005436                TST     PATRN           ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
5532  024240  001007                        BNE     3$              ;BR IF NOT
5533  024242  012700  005556                MOV     #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
5534  024246  012746  000400                MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
5535  024252  012701  052476                MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
5536  024256  000422                        BR      30$             ;PROCEED
5537  024260  005000                3$:     CLR     R0              ;INIT PATTERN NO.
5538  024262  032701  000001        4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
5539  024266  001003                        BNE     6$              ;BR IF THIS BIT IS SET
5540  024270  005200                        INC     R0              ;INCREMENT PATTERN NUMBER
5541  024272  006201                        ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
5542  024274  000772                        BR      4$              ;BR TO CHECK NEXT BIT
5543  024276  006300                6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
5544  024300  006300                        ASL     R0
5545  024302  006300                        ASL     R0
5546  024304  006300                        ASL     R0
5547  024306  006300                        ASL     R0
5548  024310  062700  005556                ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
5549  024314  012746  000020                MOV     #16.,-(SP)      ;PATTERN WORD COUNT
5550  024320  013701  005474                MOV     PMA,R1          ;SET BUFFER ADDRESS
5551  024324  010003                30$:    MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
5552  024326  011604                        MOV     (SP),R4         ;INIT PATTERN WORD COUNT
5553  024330  022321                34$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
5554  024332  001002                        BNE     31$             ;BR IF COMPARE ERROR
5555  024334  000137  024674                JMP     40$             ;JUMP IF DATA COMPARES OK
5556  024340  105037  051021        31$:    CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
5557  024344  012737  000005  052370        MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
5558                                ;COMMON COMPARE ERROR HANDLER
5559  024352  010237  001202        35$:    MOV     R2,$REG10       ;GET WORD NO.
5560  024356  163737  005456  001202        SUB     WDSXFR,$REG10
5561  024364  013737  001202  005464        MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
5562  024372  004737  023700                JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
5563  024376  104407                        SAVREG                  ;SAVE R0-R5
5564  024400  013700  005460                MOV     FINCYL,R0       ;GET CYL
5565  024404  113701  005462                MOVB    FINTRK,R1       ;GET TRACK
5566  024410  113702  005463                MOVB    FINSEC,R2       ;GET SECTOR
5567  024414  004737  027176                JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
5568  024420  104410                        RESREG                  ;RESTORE R0-R5
5569  024422  032737  001000  005414        BIT     #BADSEC,RECODE
5570  024430  001116                        BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
5571  024432  005737  005434                TST     SCRACH          ;CHECK THE ERROR COUNT
5572  024436  001010                        BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
5573  024440  104034                46$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
5574  024442  012737  177777  001174  48$:    MOV     #-1,$REG5       ;INIT CYL NO.
5575  024450  005037  001176                CLR     $REG6
5576  024454  005037  001200                CLR     $REG7
5577  024460  005237  005434        36$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
5578  024464  032777  000001  154446        BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
5579  024472  001004                        BNE     38$             ;BR TO REPORT ALL ERRORS
5580  024474  022737  000012  005434        CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
5581  024502  002504                        BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
5582  024504  023737  001174  005460  38$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
5583  024512  001010                        BNE     42$             ;BR IF YES
5584  024514  123737  001176  005462        CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
5585  024522  001004                        BNE     42$             ;BR IF YES
```

```
5586  024524  123737  001200  005463          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
5587  024532  001412                          BEQ     44$             ;BR IF SAME PACK ADDRESS
5588  024534  013737  005460  001174  42$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
5589  024542  113737  005462  001176          MOVB    FINTRK,$REG6
5590  024550  113737  005463  001200          MOVB    FINSEC,$REG7
5591  024556  104115                          ERROR   115             ;TYPE NEW PACK ADDRESS
5592  024560  005437  001202          44$:    NEG     $REG10          ;GET WORD NO.
5593  024564  016337  177776  001204          MOV     -2(R3),$REG11   ;GET GOOD DATA
5594  024572  016137  177776  001206          MOV     -2(R1),$REG12   ;GET BAD DATA
5595  024600  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
5596  024606  005037  001210                  CLR     $REG13
5597  024612  006137  001212                  ROL     $REG14
5598  024616  006137  001210                  ROL     $REG13
5599  024622  063737  005474  001212          ADD     PMA,$REG14
5600  024630  005537  001210                  ADC     $REG13
5601  024634  063737  005476  001210          ADD     PMA+2,$REG13
5602  024642  010137  001214                  MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
5603  024646  162737  000002  001214          SUB     #2,$REG15
5604  024654  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
5605  024656  032777  000100  154254          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
5606  024664  001013                          BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
5607  024666  005737  005436          50$:    TST     PATRN           ;SEE IF DEFAULT DATA TEST
5608  024672  001400                          BEQ     40$             ;BR IF YES
5609  024674  005302                  40$:    DEC     R2              ;DECREMENT WORD COUNTER
5610  024676  001406                          BEQ     54$             ;BR IF ALL DONE
5611  024700  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
5612  024702  001002                          BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
5613  024704  000137  024324                  JMP     30$             ;JUMP TO REPEAT THE PATTERN
5614  024710  000137  024330          53$:    JMP     34$
5615  024714                          54$:
5616  024714  005726                  56$:    TST     (SP)+           ;POP THE STACK
5617  024716  104410                          RESREG                  ;RESTORE R0-R5
5618  024720  000207                          RTS     PC              ;RETURN
5619
5620
5621
5622                                  ;;*****************************************************************
5623                                  ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
5624                                  ;*IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (↑Z)
5625                                  ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
5626                                  ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
5627                                  ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
5628                                  ;*NO ACTION IS TAKEN.
5629                                  ;*  CALL -   JSR     PC,CTLOUT
5630                                  ;*    OR -   CKEXIT
5631                                  ;;*****************************************************************
5632  024722  122737  000001  003117  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
5633  024730  001075                          BNE     10$             ;BR IF NOT RUNNING TESTS
5634  024732  005737  005432                  TST     INTCHR          ;SEE IF ANY TTY INPUT
5635  024736  001472                          BEQ     10$             ;BR IF NO INPUT
5636  024740  105737  003116                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
5637  024744  001441                          BEQ     12$             ;BR IF YES
5638  024746  122737  000003  005432          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
5639  024754  001026                          BNE     4$              ;BR IF NOT (↑C)
5640  024756  012700  012546                  MOV     #ALLSYS,R0      ;SET RETURN ADDR = ALLSYS
5641  024762                          2$:
```

```
5642  024762  000005                  6$:     RESET                   ;RESET ALL DEVICES
5643  024764  005037  005432                  CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
5644  024770  105037  003117                  CLRB    TSTING          ;CLEAR TTY ESCAPE ALLOW FLAG
5645  024774  012737  030640  003046          MOV     #ERRHDL,@#ABNL  ;RESTORE ERROR HANDLER ADDRESS
5646  025002  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK
5647  025006  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
5648  025014  004737  027302                  JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
5649  025020  105037  003133                  CLRB    TSTTYP          ;CLEAR OFFSET TEST IDENTIFIER
5650  025024  005037  001102                  CLR     $TSTNM          ;CLEAR THE TEST NO.
5651  025030  000110                          JMP     @R0             ;EXIT FROM TESTS
5652  025032  122737  000032  005432  4$:     CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
5653  025040  001021                          BNE     7$              ;BR IF NOT (↑Z)
5654  025042  012700  013212                  MOV     #ASKSYS,R0      ;SET RETURN ADDR = ASKSYS
5655  025046  000745                          BR      2$              ;TAKE EXIT
5656  025050  122737  000003  005432  12$:    CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
5657  025056  001012                          BNE     7$              ;BR IF NOT
5658  025060  104401  011337                  TYPE    ,HLTRQD         ;TYPE "HALT REQUESTED"
5659  025064  104401  011307                  TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN RDY"
5660  025070  012700  032676                  MOV     #HLTPRG,R0      ;SET HALT ADDRESS
5661  025074  112737  000061  003145          MOVB    #'1,PASSNO      ;RESTORE PASS NO. TO 1
5662  025102  000727                          BR      2$              ;TAKE EXIT
5663  025104  122737  000007  005432  7$:     CMPB    #007,INTCHR     ;SEE IF (↑G) TYPED
5664  025112  001002                          BNE     8$              ;BR IF NOT (↑G)
5665  025114  004737  020564                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
5666  025120  004737  020422          8$:     JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
5667  025124  000207                  10$:    RTS     PC              ;RETURN
5668
5669
5670
5671                                  ;;*****************************************************************
5672                                  ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO THE DISK.
5673                                  ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PACK
5674                                  ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
5675                                  ;*USED.
5676                                  ;;*****************************************************************
5677  025126  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
5678                                  ;LOAD THE R/W BUFFER WITH DATA
5679  025130  012700  052476                  MOV     #RWBUF,R0       ;BUFFER ADDRESS
5680  025134  012701  000400                  MOV     #400,R1         ;400(OCT) WORDS
5681  025140  010320                  4$:     MOV     R3,(R0)+        ;LOAD A BUFFER WORD
5682  025142  005301                          DEC     R1              ;DECR COUNTER
5683  025144  001375                          BNE     4$              ;BR IF NOT DONE YET
5684                                  ;SET UP PARAMETERS
5685  025146  012765  052476  000010          MOV     #RWBUF,P.BAL0(R5) ;SET BUS ADDRESS
5686  025154  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
5687                                  ;WRITE THE DATA
5688  025162  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
5689  025170  004737  027302                  JSR     PC,DRVCAL       ;WRITE THE DATA
5690  025174  104410                          RESREG                  ;RESTORE R0-R5
5691  025176  000207                          RTS     PC              ;RETURN
5692
5693
5694
5695                                  ;;*****************************************************************
5696                                  ;*LDMEM1 - LOAD MEMORY WITH INCREASING NUMBERS, STARTING WITH 1.
5697                                  ;*LDMEM2 - LOAD MEMORY WITH REPEATED WORD, IN R3 ON ENTRY.
```

```
                                ;*BOTH SUBROUTINES REQUIRE MEMORY ADDRESS IN PMA, PMA+2, AND WORD COUNT
                                ;*MUST BE IN WDSXFR.
                                ;;*****************************************************************
025200  104407                  LDMEM1: SAVREG                  ;SAVE R0-R5
025202  005004                          CLR     R4              ;R4=0 INDICATES LDMEM1
025204  012703  000001                  MOV     #1,R3           ;INIT DATA WORD TO 1
025210  000403                          BR      LDM0            ;PROCEED
025212  104407                  LDMEM2: SAVREG                  ;SAVE R0-R5
025214  012704  177777                  MOV     #-1,R4          ;R4=177777 INDICATES LDMEM2
025220  013702  005456          LDM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
025224  013701  005474                  MOV     PMA,R1          ;GET MEM ADRS
025230  010321                  6$:     MOV     R3,(R1)+        ;LOAD A WORD INTO BUFFER
025232  005704                          TST     R4              ;SEE WHICH DATA DESIRED
025234  001001                          BNE     8$              ;BR IF NOT INCREMENTING DATA
025236  005203                          INC     R3              ;INCREMENT THE DATA
025240                          8$:
025240  005302                  10$:    DEC     R2              ;DECREMENT COUNTER
025242  001372                          BNE     6$              ;BR IF NOT DONE LOADING YET
025244  104410                  12$:    RESREG                  ;RESTORE R0-R5
025246  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*CKMEM1 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH INCREASING
                                ;*NUMBERS, STARTING WITH 1.
                                ;*CKMEM2 - PERFORM SOFTWARE COMPARE OF MEMORY, WITH REPEATED DATA WORD
                                ;*IN R3 ON ENTRY.
                                ;*BOTH SUBROUTINES REQUIRE MEM ADRS IN PMA, PMA+2, AND WORD COUNT MUST BE
                                ;*IN WDSXFR.
                                ;;*****************************************************************
025250  104407                  CKMEM1: SAVREG                  ;SAVE R0-R5
025252  005004                          CLR     R4              ;R4=0 INDICATES CKMEM1
025254  012703  000001                  MOV     #1,R3           ;INIT DATA TO 1
025260  000403                          BR      CKM0            ;PROCEED
025262  104407                  CKMEM2: SAVREG                  ;SAVE R0-R5
025264  012704  177777                  MOV     #-1,R4          ;R4=177777 INDICATES CKMEM2
025270  013702  005456          CKM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
025274  013701  005474                  MOV     PMA,R1          ;GET MEM ADRS
025300  005037  005434                  CLR     SCRACH
025304  112737  000040  051021          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
025312  012737  000007  052370          MOV     #7,DF25+2
025320  004737  030360                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
025324  020321                  6$:     CMP     R3,(R1)+        ;COMPARE A DATA WORD
025326  001552                          BEQ     16$             ;BR IF DATA COMPARES OK
                                ;COMPARE ERROR HANDLER
025330  010237  001202                  MOV     R2,$REG10       ;GET WORD NO.
025334  163737  005456  001202          SUB     WDSXFR,$REG10
025342  013737  001202  005464          MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
025350  004737  023700                  JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
025354  104407                          SAVREG                  ;SAVE R0-R5
025356  013700  005460                  MOV     FINCYL,R0       ;GET CYL
025362  113701  005462                  MOVB    FINTRK,R1       ;GET TRACK
025366  113702  005463                  MOVB    FINSEC,R2       ;GET SECTOR
025372  004737  027176                  JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
025376  104410                          RESREG                  ;RESTORE R0-R5
```

```
5754  025400  032737  001000  005414          BIT     #BADSEC,RECODE
5755  025406  001122                          BNE     16$             ;BR IF LISTED- DON'T REPORT ERROR
5756  025410  105037  051021                  CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
5757  025414  012737  000005  052370          MOV     #5,DF25+2       ;ADJ. ERROR DATA WORD COUNT
5758  025422  005737  005434          9$:     TST     SCRACH          ;SEE IF FIRST ERROR IN THIS BLOCK
5759  025426  001010                          BNE     10$             ;BR IF NOT FIRST ERROR
5760  025430  104034                  17$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
5761  025432  012737  177777  001174  11$:    MOV     #-1,$REG5       ;INIT CYL NO.
5762  025440  005037  001176                  CLR     $REG6
5763  025444  005037  001200                  CLR     $REG7
5764  025450  005237  005434          10$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
5765  025454  032777  000001  153456          BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
5766  025462  001004                          BNE     12$             ;BR TO REPORT ALL ERRORS
5767  025464  022737  000012  005434          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
5768  025472  002477                          BLT     21$             ;BR IF ERROR LIMIT EXCEEDED
5769  025474  023737  001174  005460  12$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
5770  025502  001010                          BNE     14$             ;BR IF YES
5771  025504  123737  001176  005462          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
5772  025512  001004                          BNE     14$             ;BR IF YES
5773  025514  123737  001200  005463          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
5774  025522  001412                          BEQ     15$             ;BR IF SAME PACK ADDRESS
5775  025524  013737  005460  001174  14$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADR FOR PRINTOUT
5776  025532  113737  005462  001176          MOVB    FINTRK,$REG6
5777  025540  113737  005463  001200          MOVB    FINSEC,$REG7
5778  025546  104115                          ERROR   115             ;TYPE NEW PACK ADRS
5779  025550  005437  001202          15$:    NEG     $REG10          ;GET WORD NO.
5780  025554  010337  001204                  MOV     R3,$REG11       ;GOOD DATA
5781  025560  016137  177776  001206          MOV     -2(R1),$REG12   ;BAD DATA
5782  025566  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
5783  025574  005037  001210                  CLR     $REG13
5784  025600  006137  001212                  ROL     $REG14
5785  025604  006137  001210                  ROL     $REG13
5786  025610  063737  005474  001212          ADD     PMA,$REG14
5787  025616  005537  001210                  ADC     $REG13
5788  025622  063737  005476  001210          ADD     PMA+2,$REG13
5789  025630  010137  001214                  MOV     R1,$REG15       ;GET VIRT ADRS FOR PRINTOUT
5790  025634  162737  000002  001214          SUB     #2,$REG15
5791  025642  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA
5792  025644  032777  000100  153266          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
5793  025652  001007                          BNE     21$             ;BR IF JUST 1
5794                          ;PROCEED WITH COMPARISONS
5795  025654  005704                  16$:    TST     R4              ;SEE WHICH DATA DESIRED
5796  025656  001001                          BNE     18$             ;BR IF NOT INCREASING DATA
5797  025660  005203                          INC     R3              ;INCREMENT THE DATA WORD
5798  025662                          18$:
5799  025662  005302                  20$:    DEC     R2              ;DECR COUNTER
5800  025664  001402                          BEQ     21$             ;BR IF DONE COMPARING
5801  025666  000137  025324                  JMP     6$              ;CONTINUE COMPARING WORDS
5802  025672                          21$:
5803  025672  104410                  22$:    RESREG                  ;RESTORE R0-R5
5804  025674  000207                          RTS     PC              ;RETURN
5805
5806
5807
5808
5809
```

```
                                        ;;*****************************************************************
                                        ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
                                        ;*INTO BSSOFT.
                                        ;;*****************************************************************
025676                                  REDBSF:
025676  004737  020616                          JSR     PC,INITSS       ;INIT THE S.S.
025702  105065  000007                          CLRB    P.CSIH(R5)      ;SET 22-SECTOR FORMAT
025706  112765  000121  000001                  MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
025714  012765  000632  000002                  MOV     #632,P.CYLN(R5) ;SET CYL = 632
025722  112765  000002  000005                  MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
025730  012703  000012                          MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
025734  105737  003124                          TSTB    FORMAT
025740  001403                                  BEQ     6$              ;BR IF 22 SECTORS
025742  105265  000004                          INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
025746  005203                                  INC     R3
025750  012765  177400  000012  6$:             MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
025756  012765  004152  000010                  MOV     #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
025764  012737  026140  003046  8$:             MOV     #BDSCHD,A.ABNL  ;SET ERROR HANDLER FOR BSF READ
025772  105037  003135                          CLRB    WCEFLG          ;INIT THE ERROR FLAG
025776  004737  027302                          JSR     PC,DRVCAL       ;READ THE FACTORY BSF
026002  105737  003135                          TSTB    WCEFLG          ;SEE IF ANY ERRORS
026006  001413                                  BEQ     12$             ;BR IF NOT
026010  062765  000002  000004                  ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
026016  126503  000004                          CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
026022  001360                                  BNE     8$              ;BR IF NOT YET
026024  004737  030360          10$:            JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
026030  104120                                  ERROR   12.             ;ABORTING- BAD BSF READ
026032  000137  032676                          JMP     HLTPRG          ;HALT- CAN'T PROCEED
026036  012765  003152  000010  12$:            MOV     #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
026044  110365  000004                          MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
026050  012703  000026                          MOV     #22.,R3         ;SET 22 SECTOR LIMIT
026054  105737  003124                          TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
026060  001402                                  BEQ     14$             ;BR IF YES
026062  012703  000023                          MOV     #19.,R3         ;SET 20 SECTOR LIMIT
026066  012737  026140  003046  14$:            MOV     #BDSCHD,A.ABNL  ;SET ERROR HANDLER FOR BSF READ
026074  105037  003135                          CLRB    WCEFLG          ;INIT THE ERROR FLAG
026100  004737  027302                          JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
026104  105737  003135                          TSTB    WCEFLG          ;SEE IF ANY ERRORS
026110  001407                                  BEQ     16$             ;BR IF NOT
026112  062765  000002  000004                  ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
026120  126503  000004                          CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
026124  003760                                  BLE     14$             ;BR IF NOT YET
026126  000736                                  BR      10$             ;REPORT ERROR AND ABORT
026130  012737  030640  003046  16$:            MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
026136  000207                                  RTS     PC              ;RETURN

                                        ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
                                        ;*POSSIBLE ERRORS IN READING BAD SECTORS.
026140  032765  130600  000034  BDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
026146  001002                          BNE     4$              ;BR IF A READ ERROR OCCURRED
026150  000137  030640                  JMP     ERRHDL          ;GO HANDLE OTHER ERROR
026154  105237  003135          4$:     INCB    WCEFLG          ;SET ERROR FLAG
026160  000137  033110                  JMP     RETNML          ;TAKE NORMAL DRIVER RETURN
```

```
                        ;;*****************************************************************
                        ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
                        ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
                        ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
                        ;*2 BSF'S.
                        ;;*****************************************************************
026164  104407          TYPBSF: SAVREG                  ;SAVE R0-R5
026166  005004                  CLR     R4              ;INIT BSF FILE INDICATOR
026170  012700  004162          MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
026174  104401  011532          TYPE    ,FACTBS         ;TYPE "FACTORY"
026200  104401  011557          TYPE    ,BDSECT         ;TYPE "BAD SECTORS"
026204  104401  050566  3$:     TYPE    ,DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
026210  104401  001325          TYPE    ,$CRLF
026214  005001                  CLR     R1              ;INIT LIMIT COUNTER
026216  005710          4$:     TST     (R0)            ;SEE IF A SECTOR LISTED HERE
026220  100007                  BPL     8$              ;BR IF YES
026222  005701                  TST     R1              ;SEE IF FILE IS EMPTY
026224  001032                  BNE     14$             ;BR IF NOT EMPTY
026226  104401  011577          TYPE    ,NOFALS         ;TYPE "NONE"
026232  104401  001325          TYPE    ,$CRLF          ;TYPE <CR>,<LF>
026236  000425                  BR      14$
026240  012046          8$:     MOV     (R0)+,-(SP)     ;GET A CYL NO.
026242  104402                  TYPOC                   ;TYPE IT
026244  104401  011717          TYPE    ,SPACE2         ;TYPE SEPARATORS
026250  011003                  MOV     (R0),R3         ;GET TRACK AND SECTOR
026252  105003                  CLRB    R3
026254  000303                  SWAB    R3              ;GET THE TRACK NO.
026256  010346                  MOV     R3,-(SP)
026260  104402                  TYPOC                   ;TYPE IT
026262  104401  011717          TYPE    ,SPACE2         ;TYPE SEPARATORS
026266  111003                  MOVB    (R0),R3         ;GET SECTOR NO.
026270  010346                  MOV     R3,-(SP)
026272  104402                  TYPOC                   ;TYPE IT
026274  104401  001325          TYPE    ,$CRLF
026300  005720                  TST     (R0)+           ;INCR THE POINTER
026302  005201                  INC     R1              ;INCR THE COUNTER
026304  020127  000176          CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
026310  002742                  BLT     4$              ;BR IF NOT YET
026312  005704          14$:    TST     R4              ;SEE IF BOTH FILES TYPED YET
026314  001006                  BNE     18$             ;BR IF YES
026316  005204                  INC     R4              ;SET INDICATOR FOR SOFT. FILE
026320  012700  003162          MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
026324  104401  011545          TYPE    ,SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
026330  000725                  BR      3$              ;GO TYPE SOFT. BSF
026332  104410          18$:    RESREG                  ;RESTORE R0-R5
026334  000207                  RTS     PC              ;RETURN


                        ;;*****************************************************************
                        ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
                        ;*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
                        ;;*****************************************************************
026336  104407          FINMEM: SAVREG                  ;SAVE R0-R5
026340  016500  000002          MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
026344  116501  000005          MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
```

```
5922  026350  116502  000004              MOVB    P.SECT(R5),R2   ;SECTOR NO.
5923  026354  005003                      CLR     R3              ;INIT SECTOR COUNT TO 0
5924  026356  026500  000030      6$:     CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
5925  026362  001006                      BNE     8$              ;BR IF NOT EQUAL
5926  026364  126501  000027              CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
5927  026370  001003                      BNE     8$              ;BR IF NOT EQUAL
5928  026372  126502  000026              CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
5929  026376  001404                      BEQ     12$             ;BR IF ADRS ARE EQUAL
5930  026400  005203              8$:     INC     R3              ;INCR SECTOR COUNT
5931  026402  004737  026422              JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
5932  026406  000763                      BR      6$              ;GO COMPARE ADDRESSES
5933  026410  000303              12$:    SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
5934  026412  010337  005456              MOV     R3,WDSXFR       ;STORE IT
5935  026416  104410                      RESREG                  ;RESTORE R0-R5
5936  026420  000207                      RTS     PC              ;RETURN
5937
5938
5939
5940                                  ;;*****************************************************************
5941                                  ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
5942                                  ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
5943                                  ;*IN R2.
5944                                  ;;*****************************************************************
5945  026422  012704  000025      INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
5946  026426  105737  003124              TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
5947  026432  001402                      BEQ     4$              ;BR IF YES
5948  026434  012704  000023              MOV     #19.,R4         ;SET SECTOR LIMIT
5949  026440  005202              4$:     INC     R2              ;INCR. SECTOR NO.
5950  026442  020204                      CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
5951  026444  003407                      BLE     6$              ;BR IF NOT
5952  026446  005002                      CLR     R2              ;SET SECTOR = 0
5953  026450  005201                      INC     R1              ;INCR. TRACK NO.
5954  026452  020127  000002              CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
5955  026456  003402                      BLE     6$              ;BR IF NOT
5956  026460  005001                      CLR     R1              ;SET TRACK = 0
5957  026462  005200                      INC     R0              ;INCR. CYLINDER NO.
5958  026464  000207              6$:     RTS     PC              ;RETURN
5959
5960
5961
5962                                  ;;*****************************************************************
5963                                  ;*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
5964                                  ;*THIS SUBROUTINE UPDATES PMA, PMA+2, P.BALO(R5),
5965                                  ;*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
5966                                  ;*TERMINATED BY A BAD SECTOR ERROR (BSE), A DATA CHECK ERROR (DCK),
5967                                  ;*OR A HEADER VRC ERROR (HVRC).
5968                                  ;;*****************************************************************
5969  026466  104407              MIDXFR: SAVREG                  ;SAVE R0-R5
5970  026470  004737  026336              JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
5971  026474  016500  000030              MOV     P.DCYL(R5),R0   ;GET CYL NO.
5972  026500  116501  000027              MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
5973  026504  116502  000026              MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
5974  026510  013703  005456              MOV     WDSXFR,R3       ;GET NO. OF WORDS TRANSFERRED
5975  026514  032737  000020  005414      BIT     #DCKERR,RECODE  ;SEE IF DATA CHECK ERROR OCCURRED
5976  026522  001004                      BNE     9$              ;BR IF YES
5977  026524  062703  000400              ADD     #400,R3         ;SKIP BAD SECTOR
```

```
 5978  026530  004737  026422                  JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
 5979  026534  010065  000002          9$:     MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
 5980  026540  110165  000005                  MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
 5981  026544  110265  000004                  MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
 5982  026550  060365  000012                  ADD     R3,P.WC(R5)     ;UPDATE P.WC(R5)
 5983  026554  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;SEE IF BUS ADRS INCR INHIB SET
 5984  026562  001022                          BNE     16$             ;BR IF YES
 5985  026564  005004                          CLR     R4              ;GET BYTES XFERRED
 5986  026566  006103                          ROL     R3
 5987  026570  006104                          ROL     R4
 5988  026572  060337  005474                  ADD     R3,PMA          ;UPDATE PMA,PMA+2
 5989  026576  005537  005476                  ADC     PMA+2
 5990  026602  060437  005476                  ADD     R4,PMA+2
 5991  026606  013765  005474  000010          MOV     PMA,P.BALO(R5)  ;UPDATE P.BALO(R5)
 5992  026614  013700  005476                  MOV     PMA+2,R0
 5993  026620  042700  177774                  BIC     #177774,R0
 5994  026624  150065  000007                  BISB    R0,P.BAHI(R5)   ;UPDATE P.BAHI(R5)
 5995  026630  104410                  16$:    RESREG                  ;RESTORE R0-R5
 5996  026632  000207                          RTS     PC              ;RETURN
 5997
 5998
 5999
 6000                                  ;;*****************************************************************
 6001                                  ;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
 6002                                  ;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
 6003                                  ;;*****************************************************************
 6004  026634  104407                  SVPRMS: SAVREG                  ;SAVE R0-R5
 6005  026636  012700  002632                  MOV     #PARM0+2,R0     ;ADRS OF PARAMS
 6006  026642  012701  005444                  MOV     #SAVPRS,R1      ;ADRS OF SAVE AREA
 6007  026646  016537  000012  005456          MOV     P.WC(R5),WDSXFR ;GET WORD COUNT
 6008  026654  005437  005456                  NEG     WDSXFR          ;MAKE IT POSITIVE
 6009  026660  012737  052476  005474          MOV     #RWBUF,PMA      ;INIT PMA TO RWBUF
 6010  026666  005037  005476                  CLR     PMA+2           ;INIT PMA+2 TO 0
 6011  026672  000405                          BR      GTO             ;CONTINUE
 6012  026674  104407                  GTPRMS: SAVREG                  ;SAVE R0-R5
 6013  026676  012700  005444                  MOV     #SAVPRS,R0      ;ADRS OF SAVE AREA
 6014  026702  012701  002632                  MOV     #PARM0+2,R1     ;ADRS OF PARAMS
 6015  026706  012702  000005          GTO:    MOV     #5,R2           ;SET FOR 5 WORDS
 6016  026712  012021                  6$:     MOV     (R0)+,(R1)+     ;MOVE A WORD
 6017  026714  005302                          DEC     R2              ;SEE IF DONE YET
 6018  026716  001375                          BNE     6$              ;BR IF NOT YET
 6019  026720  104410                          RESREG                  ;RESTORE R0-R5
 6020  026722  000207                          RTS     PC              ;RETURN
 6021
 6022
 6023
 6024                                  ;;*****************************************************************
 6025                                  ;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS
 6026                                  ;*OR DATA CHECK ERRORS, IF ENCOUNTERED.
 6027                                  ;*THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
 6028                                  ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED,
 6029                                  ;*AND IT IS ALSO USED TO HANDLE THE DATA CHECK ERRORS CAUSED
 6030                                  ;*BY READING DATA WITH OFFSETS, IN THE OVERWRITE AND DATA COMPATIBILITY
 6031                                  ;*TESTS.
 6032                                  ;;*****************************************************************
 6033  026724  010446                  TRNSFR: MOV     R4,-(SP)        ;SAVE R4 ON STACK
```

```
 6034  026726  005004                        CLR     R4              ;CLEAR ERROR INDICATOR
 6035  026730  105037  003134                CLRB    DCEFLG          ;CLEAR DCK ERROR FLAG
 6036  026734  005037  005414          4$:   CLR     RECODE          ;CLEAR ERROR FLAGS
 6037  026740  012737  020204  003046        MOV     #DCKHDL,A.ABNL  ;SET DCK ERROR HANDLER ADRS
 6038  026746  105037  003125                CLRB    ERRCNT          ;CLEAR ERROR RETRY COUNT
 6039  026752  004737  027302                JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
 6040  026756  032737  000026  005414        BIT     #BSERR!DCKERR!HVRCER,RECODE  ;SEE IF BSE OR DCK OR HVRC ERROR OCCURRED
 6041  026764  001410                        BEQ     6$              ;BR IF NOT
 6042  026766  005204                        INC     R4              ;INCR ERROR INDICATOR
 6043  026770  004737  022602                JSR     PC,CKSCOR       ;DECR. DRIVE SCORE, IF NECESSARY
 6044  026774  004737  026466                JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
 6045  027000  005765  000012                TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
 6046  027004  001353                        BNE     4$              ;BR IF NOT
 6047  027006  005704                6$:     TST     R4              ;SEE IF ANY ERRORS OCCURRED
 6048  027010  001404                        BEQ     8$              ;BR IF NOT
 6049  027012  004737  026674                JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
 6050  027016  004737  026634                JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
 6051  027022  012604                8$:     MOV     (SP)+,R4        ;RESTORE R4
 6052  027024  012737  030640  003046        MOV     #ERRHDL,A.ABNL  ;RESTORE NORMAL ERROR HANDLER ADRS
 6053  027032  000207                        RTS     PC              ;RETURN
 6054
 6055
 6056
 6057                                ;;*****************************************************************
 6058                                .SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
 6059                                ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
 6060                                ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
 6061                                ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
 6062                                ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
 6063                                ;*
 6064                                ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
 6065                                ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
 6066                                ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
 6067                                ;;*****************************************************************
 6068  027034  010237  001266        $RHTBS: MOV     R2,$TMP2
 6069  027040  010337  001270                MOV     R3,$TMP3
 6070  027044  012637  001272                MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
 6071  027050  011402                        MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
 6072  027052  012437  001264                MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
 6073  027056  062737  001000  001264        ADD     #1000,$TMP1
 6074  027064  005003                        CLR     R3              ;CLEAR R3 FOR COUNT
 6075  027066  062702  000010                ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
 6076  027072  005712                1$:     TST     (R2)            ;TEST IF ALL ONES
 6077  027074  100430                        BMI     5$              ;YES-DONE
 6078  027076  020012                        CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
 6079  027100  001406                        BEQ     3$              ;YES-GO CHECK TRACK
 6080  027102  062702  000004                ADD     #4,R2           ;ELSE BUMP POINTER
 6081  027106  020237  001264                CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
 6082  027112  002021                        BGE     5$              ;YES-EXIT
 6083  027114  000766                        BR      1$              ;LOOP
 6084  027116  005722                3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
 6085  027120  011237  001302                MOV     (R2),$TMP10     ;GET TRK/SEC WORD
 6086  027124  042737  174377  001302        BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
 6087  027132  123701  001303                CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
 6088  027136  001402                        BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
 6089  027140  005722                        TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
```

```
 6090  027142  000753                    BR      1$              ;GO TEST NEXT CYL
 6091  027144  005203            4$:     INC     R3              ;BUMP BAD SECTOR COUNT
 6092  027146  012246                    MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
 6093  027150  042716  177700            BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
 6094  027154  000746                    BR      1$              ;GO CHECK REST OF FILE
 6095  027156  010346            5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
 6096  027160  013702  001266            MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
 6097  027164  013703  001270            MOV     $TMP3,R3        ;REGISTERS
 6098  027170  013746  001272            MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
 6099  027174  000204                    RTS     R4              ;RETURN
 6100                                ;;*********************************************************************
 6101                                .SBTTL  BAD SECTOR CHECK ROUTINE
 6102                                ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
 6103                                ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
 6104                                ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
 6105                                ;*IS NOT LISTED THE FLAG IS RESET.
 6106                                ;;*********************************************************************
 6107  027176  104407            BDSRCK: SAVREG
 6108  027200  012737  004152  027212    MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
 6109  027206  004437  027034    2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
 6110  027212  000000    1$:             .WORD                   ;TABLE ADDRESS GOES HERE
 6111  027214  012603                    MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
 6112  027216  001015                    BNE     6$              ;IF ANY, GO TEST WHICH ONES
 6113  027220  023727  027212  003152  7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
 6114  027226  001404                    BEQ     3$              ;IF YES-EXIT
 6115  027230  012737  003152  027212    MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
 6116  027236  000763                    BR      2$              ;GO SEARCH IT
 6117  027240  042737  001000  005414  3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
 6118  027246  104410            4$:     RESREG
 6119  027250  000207                    RTS     PC              ;RETURN
 6120  027252  022602            6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
 6121  027254  001403                    BEQ     8$              ;YES-GO SET BIT & EXIT
 6122  027256  005303                    DEC     R3              ;DECREMENT BAD SECTOR COUNT
 6123  027260  001374                    BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
 6124  027262  000756                    BR      7$              ;ELSE GO SEARCH OTHER TABLE
 6125  027264  052737  001000  005414  8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
 6126  027272  005303            9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
 6127  027274  001764                    BEQ     4$              ;NUMBER
 6128  027276  005726                    TST     (SP)+
 6129  027300  000774                    BR      9$              ;EXIT
 6130
 6131
 6132
 6133
 6134                                ;;*********************************************************************
 6135                                .SBTTL  CALL DRIVER ROUTINE
 6136                                ;*ENTRY JSR PC,DRVCAL
 6137                                ;*              WITH R5 POINTING TO PARAMETER BLOCK
 6138                                ;*RETURN RTS PC
 6139                                ;*
 6140                                ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
 6141                                ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
 6142                                ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
 6143                                ;*BLOCK WHEN THE ROUTINE IS CALLED.
 6144                                ;*
 6145                                ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
```

```
                                ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
                                ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
                                ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
                                ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
                                ;;*****************************************************************
027302                          DRVCAL:
027302  004737  023556                JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
027306  004737  024722                JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
027312  105037  003122                CLRB    DONE            ;CLEAR DONE FLAG
027316  105037  003131                CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
027322  004737  027354                JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
027326  010537  027336                MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
027332  004737  037356                JSR     PC,C.INIT       ;CALL DRIVER
027336  000000                  4$:   .WORD                   ;P.B. ADDRESS GOES HERE
027340  004737  033726          6$:   JSR     PC,W.WTCH       ;CALL WATCH DOG
027344  105737  003122                TSTB    DONE            ;DONE SET?
027350  001773                        BEQ     6$              ;NO-LOOP
027352  000207                        RTS     PC              ;YES-RETURN


                                ;;*****************************************************************
                                ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
                                ;;*****************************************************************
027354  104407                  STRCMD: SAVREG                  ;SAVE R0-R5
027356  012701  005152                MOV     #PRVCMD,R1      ;ADDR OF PREV COMMAND STORAGE
027362  012700  005166                MOV     #COMSTR,R0      ;ADDR OF CURRENT COMMAND STORAGE
                                ;STORE PREVIOUS COMMAND
027366  012021                        MOV     (R0)+,(R1)+
027370  012021                        MOV     (R0)+,(R1)+
027372  012021                        MOV     (R0)+,(R1)+
027374  012021                        MOV     (R0)+,(R1)+
027376  012021                        MOV     (R0)+,(R1)+
027400  012021                        MOV     (R0)+,(R1)+
                                ;STORE CURRENT COMMAND
027402  012701  005166          4$:   MOV     #COMSTR,R1
027406  012521                        MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BLK
027410  012521                        MOV     (R5)+,(R1)+
027412  012521                        MOV     (R5)+,(R1)+
027414  012521                        MOV     (R5)+,(R1)+
027416  012521                        MOV     (R5)+,(R1)+
027420  012521                        MOV     (R5)+,(R1)+
027422  104410                        RESREG                  ;RESTORE R0-R5
027424  000207                        RTS     PC              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
                                ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
                                ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
                                ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
                                ;*ROUTINE.
                                ;;*****************************************************************
027426  152737  000377  003122  ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
027434  032737  100000  005414          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
```

```
 6202  027442  001403                        BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
 6203  027444  105037  003125                CLRB    ERRCNT          ;CLEAR ERROR COUNT
 6204  027450  000413                        BR      1$              ;EXIT
 6205  027452  105737  003125          2$:   TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
 6206  027456  001410                        BEQ     1$              ;NO - SKIP TO EXIT
 6207  027460  005037  001174                CLR     $REGS
 6208  027464  113737  003125  001174        MOVB    ERRCNT,$REGS    ;GET RETRY COUNT
 6209  027472  104101                        ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
 6210  027474  105037  003125                CLRB    ERRCNT          ;CLEAR ERROR COUNT
 6211  027500  005037  005414          1$:   CLR     RECODE          ;CLEAR RECOVERY FLAGS
 6212  027504  000207                        RTS     PC              ;RETURN
 6213                                  ;;*****************************************************************
 6214                                  .SBTTL  TYPE ERROR ROUTINE
 6215                                  ;*ENTRY -       JSR     PC,TYPERR
 6216                                  ;*RETURN -      RTS     PC
 6217                                  ;*
 6218                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 6219                                  ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
 6220                                  ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
 6221                                  ;*THE ERROR.
 6222                                  ;;*****************************************************************
 6223  027506  104407                  TYPERR: SAVREG
 6224  027510  105237  003127                  INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
 6225  027514  032777  020000  151416  9$:     BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
 6226  027522  001402                        BEQ     6$              ;BR IF NO
 6227  027524  000137  030130                JMP     20$
 6228  027530  005000                  6$:   CLR     R0              ;CLR R0 FOR ERROR NUMBER
 6229  027532  005005                        CLR     R5              ;INIT INDENT INDICATOR
 6230  027534  005105                        COM     R5
 6231  027536  113700  001114                MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
 6232  027542  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
 6233  027544  006300                        ASL     R0
 6234  027546  006300                        ASL     R0
 6235  027550  006300                        ASL     R0
 6236  027552  062700  001410          1$:   ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
 6237  027556  012037  027576                MOV     (R0)+,2$        ;GET EM POINTER
 6238  027562  001406                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 6239  027564  104401  011703                TYPE    ,CR2LF
 6240  027570  104401  044546                TYPE    ,AS2SP2         ;TYPE "?? "
 6241  027574  104401                        TYPE                    ;TYPE ERROR MESSAGE (EM)
 6242  027576  000000                  2$:   .WORD   0               ;EM POINTER GOES HERE
 6243  027600  012037  027754          3$:   MOV     (R0)+,4$        ;GET DH POINTER
 6244  027604  001467                        BEQ     5$              ;BR IF THERE ISN'T ONE
 6245  027606  104401  001325                TYPE    ,$CRLF
 6246  027612  104401  044534                TYPE    ,TSTMSG         ;TYPE "    TEST "
 6247  027616  013746  001102                MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
 6248  027622  104403                        TYPOS                   ;TYPE IT
 6249  027624     002                        .BYTE   2               ;2 DIGITS
 6250  027625     000                        .BYTE   0               ;SUPPRESS LEADING ZEROS
 6251  027626  104401  011703                TYPE    ,CR2LF
 6252  027632  032777  010000  151300        BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 6253  027640  001133                        BNE     20$             ;BR IF YES
 6254  027642  104401  047670                TYPE    ,DH105          ;TYPE "PREVIOUS COMMAND :"
 6255  027646  104401  001325                TYPE    ,$CRLF
 6256  027652  104401  050062                TYPE    ,DH101+10       ;TYPE PREV COMMAND HEADER
 6257  027656  104401  001325                TYPE    ,$CRLF
```

```
6258  027662  012701  000006                    MOV     #6,R1           ;SIX COMMAND VALUES
6259  027666  012702  001236                    MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
6260  027672  012246                  30$:      MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
6261  027674  104402                            TYPOC                   ;TYPE IT
6262  027676  104401  011717                    TYPE    ,SPACE2         ;TYPE SEPARATORS
6263  027702  005301                            DEC     R1              ;SEE IF 7 VALUES TYPED YET
6264  027704  001372                            BNE     30$             ;BR IF NOT
6265  027706  104401  001325                    TYPE    ,$CRLF
6266  027712  104401  011717                    TYPE    ,SPACE2         ;INDENT
6267  027716  104401  050141                    TYPE    ,DH102          ;TYPE HDR FOR BA DATA
6268  027722  104401  001325                    TYPE    ,$CRLF
6269  027726  104401  011717                    TYPE    ,SPACE2         ;INDENT
6270  027732  012246                            MOV     (R2)+,-(SP)
6271  027734  104402                            TYPOC                   ;TYPE PREV. HI BA BITS
6272  027736  104401  011717                    TYPE    ,SPACE2
6273  027742  011246                            MOV     (R2),-(SP)
6274  027744  104402                            TYPOC                   ;TYPE PREV. LO BA BITS
6275  027746  104401  011703                    TYPE    ,CR2LF
6276  027752  104401                            TYPE                    ;TYPE DATA HEADER
6277  027754  000000                  4$:       .WORD   0               ;DH POINTER GOES HERE
6278  027756  104401  001325                    TYPE    ,$CRLF
6279  027762  005005                            CLR     R5              ;INIT INDENT INDICATOR
6280  027764  032777  010000  151146  5$:       BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
6281  027772  001056                            BNE     20$             ;BR IF YES
6282  027774  012001                            MOV     (R0)+,R1        ;GET DT POINTER
6283  027776  001454                            BEQ     20$             ;BRANCH IF THERE ARE NONE
6284  030000  012000                            MOV     (R0)+,R0        ;GET DF POINTER
6285  030002  012002                            MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
6286  030004  112003                  10$:      MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
6287  030006  105720                            TSTB    (R0)+           ;BUMP PAST FORMAT WORD
6288  030010  005703                            TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
6289  030012  001417                            BEQ     14$             ;NO - SKIP DATA PRINT
6290  030014  005705                            TST     R5              ;SEE IF SHOULD INDENT
6291  030016  001002                            BNE     11$             ;BR IF NOT
6292  030020  104401  011717                    TYPE    ,SPACE2         ;INDENT
6293  030024  013146                  11$:      MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
6294  030026  104402                            TYPOC                   ;TYPE IT
6295  030030  005303                            DEC     R3              ;MORE DATA WORDS
6296  030032  001403                            BEQ     12$             ;NO-BRANCH
6297  030034  104401  011717                    TYPE    ,SPACE2         ;TYPE SEPARATORS
6298  030040  000771                            BR      11$             ;LOOP
6299  030042  005702                  12$:      TST     R2              ;SEE IF <CR>,<LF> NEEDED
6300  030044  001402                            BEQ     14$             ;BR IF NOT
6301  030046  104401  001325                    TYPE    ,$CRLF          ;TYPE IT
6302  030052  005302                  14$:      DEC     R2              ;MORE DH'S?
6303  030054  003425                            BLE     20$             ;NO-BRANCH
6304  030056  012037  030110          15$:      MOV     (R0)+,16$       ;GET NEXT DH POINTER
6305  030062  105710                            TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
6306  030064  001004                            BNE     34$             ;BR IF YES
6307  030066  104401  001325                    TYPE    ,$CRLF          ;SKIP EXTRA LINE
6308  030072  005005                            CLR     R5              ;RE-INIT INDENT INDICATOR
6309  030074  000404                            BR      36$
6310  030076  005105                  34$:      COM     R5              ;COMPLEMENT INDENT INDICATOR
6311  030100  001002                            BNE     36$             ;BR IF NO INDENT REQUIRED
6312  030102  104401  011717                    TYPE    ,SPACE2         ;INDENT
6313  030106  104401                  36$:      TYPE                    ;TYPE DH
```

```
 6314  030110  000000                    16$:    .WORD   0               ;DH POINTER GOES HERE
 6315  030112  104401  001325                    TYPE    $CRLF
 6316  030116  105710                            TSTB    (R0)            ;TYPE A DT?
 6317  030120  001331                            BNE     10$             ;YES-BRANCH
 6318  030122  062700  000002                    ADD     #2,R0           ;INCREMENT DF POINTER
 6319  030126  000751                            BR      14$             ;SEE IF END OF DF BLOCK
 6320  030130  104410                    20$:    RESREG
 6321  030132  000207                            RTS     PC
 6322                                    ;;*****************************************************************
 6323                                    .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
 6324                                    ;*ENTRY:        JSR PC, CONERR
 6325                                    ;*RETURN:       RTS PC
 6326                                    ;*
 6327                                    ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
 6328                                    ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
 6329                                    ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
 6330                                    ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
 6331                                    ;*AT THIS TIME.
 6332                                    ;;*****************************************************************
 6333  030134  104407                    CONERR: SAVREG                  ;SAVE R0-R5
 6334  030136  152737  000377  003122            BISB    #377,DONE       ;SET DONE FLAG
 6335  030144  105237  003125                    INCB    ERRCNT          ;INCREMENT ERROR COUNT
 6336  030150  004737  033120                    JSR     PC,TOPROC       ;LOAD RK REGS INTO $REGS
 6337  030154  032737  000001  003052            BIT     #BIT0,E.CONT    ;ERROR 0?
 6338  030162  001402                            BEQ     1$              ;NO-BRANCH
 6339  030164  104064                            ERROR   64              ;CLEAR CONT DID NOT CLEAR ERROR
 6340  030166  000470                            BR      7$
 6341  030170  032737  000002  003052    1$:     BIT     #BIT1,E.CONT    ;ERROR 1?
 6342  030176  001402                            BEQ     2$              ;NO-BRANCH
 6343  030200  104065                            ERROR   65              ;NO ATTENTION IN ATTENTION SUM REG
 6344  030202  000462                            BR      7$
 6345  030204  032737  000004  003052    2$:     BIT     #BIT2,E.CONT    ;ERROR 2?
 6346  030212  001407                            BEQ     3$              ;NO-BRANCH
 6347  030214  105737  003131                    TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
 6348  030220  001402                            BEQ     15$             ;BR IF NOT
 6349  030222  105237  003132                    INCB    REISSU          ;SET FLAG TO RE-ISSUE COMMAND
 6350  030226  104066                    15$:    ERROR   66              ;UNSOLICITED ATTENTION
 6351  030230  000447                            BR      7$
 6352  030232  032737  000010  003052    3$:     BIT     #BIT3,E.CONT    ;ERROR 3?
 6353  030240  001402                            BEQ     4$              ;NO-BRANCH
 6354  030242  104067                            ERROR   67              ;UNEXPECTED DATA TYPE ERROR
 6355  030244  000441                            BR      7$
 6356  030246  032737  000020  003052    4$:     BIT     #BIT4,E.CONT    ;ERROR 4?
 6357  030254  001402                            BEQ     5$              ;NO-BRANCH
 6358  030256  104070                            ERROR   70              ;ATTENTION DID NOT RESET WITH CLEAR
 6359  030260  000433                            BR      7$
 6360  030262  032737  000040  003052    5$:     BIT     #BIT5,E.CONT    ;ERROR 5?
 6361  030270  001402                            BEQ     6$              ;NO-BRANCH
 6362  030272  104071                            ERROR   71              ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
 6363  030274  000425                            BR      7$
 6364  030276  032737  000400  003052    6$:     BIT     #BIT8,E.CONT
 6365  030304  001401                            BEQ     8$
 6366  030306  104072                            ERROR   72              ;DATA LATE WHEN UNLOADING HEADER
 6367  030310  032737  001000  003052    8$:     BIT     #BIT9,E.CONT
 6368  030316  001401                            BEQ     9$
 6369  030320  104073                            ERROR   73              ;CONTROLLER ERROR DURING DRIVER SERVICE
```

```
6370  030322  032737  002000  003052  9$:     BIT     #BIT10,E.CONT
6371  030330  001401                          BEQ     10$
6372  030332  104074                          ERROR   74              ;DRIVE DETECTED PARITY ERROR
6373  030334  032737  100000  003052  10$:    BIT     #BIT15,E.CONT
6374  030342  001401                          BEQ     11$
6375  030344  104052                          ERROR   52              ;MULTIPLE DRIVE SELECT
6376  030346  104075                  11$:    ERROR   75              ;UNDEFINED ERROR
6377  030350  005037  003052          7$:     CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
6378  030354  000137  032714                  JMP     BGNRTY          ;GO DO RETRY
6379                                  ;;*****************************************************************
6380                                  .SBTTL  REPORT SUPPORT ROUTINE
6381                                  ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
6382                                  ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING.  ALL THESE MAY
6383                                  ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
6384  030360                          REPSUP:
6385  030360  104407                          SAVREG
6386  030362  005037  005416                  CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
6387  030366  116537  000001  005416          MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
6388  030374  012700  001162                  MOV     #$REG0,R0       ;FOR REPORTING
6389  030400  016520  000002                  MOV     P.CYLN(R5),(R0)+
6390  030404  116520  000005                  MOVB    P.TRCK(R5),(R0)+
6391  030410  105020                          CLRB    (R0)+
6392  030412  116520  000004                  MOVB    P.SECT(R5),(R0)+
6393  030416  105020                          CLRB    (R0)+
6394  030420  016520  000012                  MOV     P.WC(R5),(R0)+
6395  030424  012700  001174                  MOV     #$REG5,R0
6396  030430  116503  000007                  MOVB    P.BAHI(R5),R3
6397  030434  042703  177774                  BIC     #177774,R3
6398  030440  010337  001256                  MOV     R3,$REG36       ;HI BA BITS
6399  030444  016537  000010  001260          MOV     P.BALO(R5),$REG37 ;LO BA BITS
6400  030452  016520  000016                  MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
6401  030456  016520  000020                  MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
6402  030462  016520  000030                  MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
6403  030466  016520  000026                  MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
6404  030472  016520  000022                  MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
6405                                                                  ;FOR ALL REPORTS (TO BE
6406  030476  016520  000024                  MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
6407  030502  016520  000032                  MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
6408  030506  016520  000036                  MOV     P.DS(R5),(R0)+
6409  030512  016520  000034                  MOV     P.ER(R5),(R0)+
6410  030516  016520  000040                  MOV     P.AOO(R5),(R0)+
6411  030522  016520  000042                  MOV     P.BOO(R5),(R0)+
6412  030526  016520  000044                  MOV     P.A01(R5),(R0)+
6413  030532  016520  000046                  MOV     P.B01(R5),(R0)+
6414  030536  016520  000050                  MOV     P.A10(R5),(R0)+
6415  030542  016520  000052                  MOV     P.B10(R5),(R0)+
6416  030546  016520  000054                  MOV     P.A11(R5),(R0)+
6417  030552  016520  000056                  MOV     P.B11(R5),(R0)+
6418                                  ;STORE PREVIOUS COMMAND FOR PRINTOUT
6419  030556  012701  001236                  MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
6420  030562  012700  005152                  MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
6421  030566  112021                          MOVB    (R0)+,(R1)+     ;DRIVE NO.
6422  030570  105021                          CLRB    (R1)+
6423  030572  112021                          MOVB    (R0)+,(R1)+     ;COMMAND
6424  030574  105021                          CLRB    (R1)+
6425  030576  012021                          MOV     (R0)+,(R1)+     ;CYL ADDRESS
```

```
 6426  030600  116021  000001                MOVB    1(R0),(R1)+     ;TRACK
 6427  030604  105021                        CLRB    (R1)+
 6428  030606  111021                        MOVB    (R0),(R1)+      ;SECTOR
 6429  030610  105021                        CLRB    (R1)+
 6430  030612  016021  000006                MOV     6(R0),(R1)+     ;WORD COUNT
 6431  030616  116003  000003                MOVB    3(R0),R3        ;HI BA BITS
 6432  030622  042703  177774                BIC     #177774,R3
 6433  030626  010321                        MOV     R3,(R1)+
 6434  030630  016011  000004                MOV     4(R0),(R1)      ;LO BA BITS
 6435  030634  104410                        RESREG
 6436  030636  000207                        RTS     PC
 6437                                  ;;*****************************************************************
 6438                                  .SBTTL  REPORT ERROR ROUTINE
 6439                                  ;* ENTRY         JSR     PC, ERRHDL
 6440                                  ;*RETURN        RTS     PC
 6441                                  ;*
 6442                                  ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
 6443                                  ;*IN THE OPERATION.  THE ROUTINE DETERMINES WHICH COMMAND WAS
 6444                                  ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
 6445                                  ;;*****************************************************************
 6446
 6447  030640  104407                ERRHDL: SAVREG
 6448  030642  152737  000377  003122        BISB    #377,DONE       ;SET DONE FLAG
 6449  030650  105237  003125                INCB    ERRCNT          ;INCREMENT ERROR COUNT
 6450  030654  005037  005414                CLR     RECODE          ;CLEAR RECOVERY CODE WORD
 6451  030660  032737  000400  005414 ER2ENT: BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
 6452  030666  001402                        BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
 6453  030670  012705  002714                MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
 6454  030674  012737  033072  003046 52$:   MOV     #RETANL,A.ABNL  ;SET NOW ABNORMAL AND NORMAL RETURN FOR
 6455  030702  012737  033110  003044        MOV     #RETNML,A.NORM  ;DRIVER OPERATIONS IN ERROR PROCESSING
 6456  030710  004737  030360                JSR     PC,REPSUP       ;GO SET UP REGISTERS FOR REPORT
 6457                                                                 ;NOW BEGIN TESTING THE ERROR
 6458                                                                 ;BITS. THE SEQUENCE IN
 6459                                                                 ;WHICH THEY ARE TESTED IS
 6460                                                                 ;CONSIDERED SIGNIFICANT IN
 6461                                                                 ;THAT ERRORS OF A MORE
 6462                                                                 ;CATASTROPHIC NATURE ARE FIRST
 6463                                                                 ;TESTED.
 6464                                                                 ;
 6465                                                                 ;IF AN ERROR IS FOUND SET,
 6466                                                                 ;THAT ERROR IS REPORTED AND
 6467                                                                 ;THE REPORTING IS TERMINATED.
 6468                                                                 ;IF ADDITIONAL ERRORS ARE SET,
 6469                                                                 ;THE RK611 REGISTER PRINTOUTS
 6470                                                                 ;WILL SHOW THIS BUT THE
 6471                                                                 ;REGISTER CONTENTS MUST BE
 6472                                                                 ;MANUALLY DECODED TO LOCATE THE
 6473                                                                 ;SECOND ERROR
 6474  030714  016504  000034                MOV     P.ER(R5),R4     ;SET R4 TO ERROR REGISTER
 6475  030720  032765  020000  000020        BIT     #UPE,P.CS2(R5)  ;TEST UPE, IF UES-SET
 6476  030726  001406                        BEQ     1$              ;REPORT ERROR
 6477  030730  104001                        ERROR   1
 6478  030732  052737  000200  005414        BIS     #ABORT,RECODE
 6479  030740  000137  032372                JMP     37$
 6480  030744  032765  004000  000020 1$:    BIT     #NEM,P.CS2(R5)  ;TEST NON-EXISTANT MEMORY
 6481  030752  001406                        BEQ     2$
```

```
6482  030754  104002                        ERROR   2
6483  030756  052737  000200  005414        BIS     #ABORT,RECODE
6484  030764  000137  032372                JMP     37$
6485  030770  032765  010000  000020  2$:   BIT     #NED,P.CS2(R5)  ;TEST NON-EXISTANT DRIVE
6486  030776  001412                        BEQ     3$
6487  031000  032765  000400  000020        BIT     #UFE,P.CS2(R5)  ;TEST IF NED & UFE BOTH SET
6488  031006  001403                        BEQ     38$
6489  031010  104035                        ERROR   35              ;NED & UFE BOTH SET ERROR
6490  031012  000137  032372                JMP     37$
6491  031016  104003              38$:      ERROR   3               ;NED ONLY
6492  031020  000137  032372                JMP     37$
6493  031024  032765  000400  000020  3$:   BIT     #UFE,P.CS2(R5)  ;TEST UNIT FIELD ERROR
6494  031032  001412                        BEQ     4$
6495  031034  032765  010000  000020        BIT     #NED,P.CS2(R5)  ;TEST IF UFE & NED BOTH SET
6496  031042  001403                        BEQ     39$             ;NO-SKIP
6497  031044  104035                        ERROR   35              ;REPORT NED & UFE BOTH SET
6498  031046  000137  032372                JMP     37$
6499  031052  104004              39$:      ERROR   4               ;REPORT UFE ONLY
6500  031054  000137  032372                JMP     37$
6501  031060  032765  000100  000014  4$:   BIT     #CMDTO,P.PRST(R5) ;
6502  031066  001423                        BEQ     5$
6503  031070  004737  033120                JSR     PC,TOPROC       ;GO PROCESS TIMEOUT
6504  031074  032737  002000  005414        BIT     #TWOTOS,RECODE  ;2ND TIMEOUT IN TIMEOUT PROC?
6505  031102  001007                        BNE     40$             ;YES - SKIP TO ERROR 56
6506  031104  032737  000400  005414        BIT     #LEV2ER,RECODE  ;TEST IF LEVEL 2 ERROR
6507  031112  001006                        BNE     41$             ;YES - SKIP TO ERROR 57
6508  031114  104005                        ERROR   5               ;ELSE MAKE FULL TIMEOUT REPORT
6509  031116  000137  032372                JMP     37$
6510  031122  104056              40$:      ERROR   56              ;TWO TIMEOUTS ERROR REPORT
6511  031124  000137  032372                JMP     37$
6512  031130  104057              41$:      ERROR   57              ;2ND LEVEL ERROR REPORT
6513  031132  000137  030660                JMP     ER2ENT          ;GO BUILD AND MAKE 2ND REPORT
6514  031136  032765  010000  000014  5$:   BIT     #DRVSZD,P.PRST(R5) ;SEE IF DRIVE SIEZED BY OTHER PORT
6515  031144  001403                        BEQ     65$             ;BR IF DRIVE IS AVAILABLE
6516  031146  104107                        ERROR   107             ;DRIVE SIEZED BY OTHER PORT
6517  031150  000137  032372                JMP     37$
6518  031154  032765  020000  000016  65$:  BIT     #SPAR,P.CS1(R5) ;TEST D TO C PARITY ERROR
6519  031162  001406                        BEQ     6$
6520  031164  104006                        ERROR   6
6521  031166  052737  004000  005414        BIS     #RCLREQ,RECODE
6522  031174  000137  032372                JMP     37$
6523  031200  032704  000010          6$:   BIT     #DRPAR,R4       ;TEST DRIVE DETECTED PARITY ERROR
6524  031204  001406                        BEQ     7$
6525  031206  104007                        ERROR   7
6526  031210  052737  004000  005414        BIS     #RCLREQ,RECODE
6527  031216  000137  032372                JMP     37$
6528  031222  032765  000010  000036  7$:   BIT     #ACLO,P.DS(R5)  ;TEST AC LOW
6529  031230  001403                        BEQ     8$
6530  031232  104010                        ERROR   10
6531  031234  000137  032372                JMP     37$
6532  031240  032765  000020  000036  8$:   BIT     #SPDLSS,P.DS(R5) ;TEST SPEED LOSS
6533  031246  001403                        BEQ     24$
6534  031250  104011                        ERROR   11
6535  031252  000137  032372                JMP     37$
6536  031256  032765  004000  000016  24$:  BIT     #CTO,P.CS1(R5)  ;TEST FOR CONTROLLER TIMEOUT
6537  031264  001401                        BEQ     25$
```

```
6538  031266  104027                          ERROR   27
6539  031270  032704  000001                25$:  BIT     #ILC,R4         ;TEST ILLEGAL FUNCTION CODE
6540  031274  001403                          BEQ     10$
6541  031276  104012                          ERROR   12
6542  031300  000137  032372                JMP     37$
6543  031304  032765  002000  000020  10$:   BIT     #PGE,P.CS2(R5)  ;TEST PROGRAMMING ERROR
6544  031312  001403                          BEQ     11$
6545  031314  104013                          ERROR   13
6546  031316  000137  032372                JMP     37$
6547  031322  032704  000004                11$:   BIT     #ILF,R4         ;TEST ILLEGAL DRIVE FUNCTION
6548  031326  001403                          BEQ     12$
6549  031330  104014                          ERROR   14
6550  031332  000137  032372                JMP     37$
6551  031336  032704  000040                12$:   BIT     #DTYE,R4        ;TEST DRIVE TYPE ERROR
6552  031342  001403                          BEQ     13$
6553  031344  104015                          ERROR   15
6554  031346  000137  032372                JMP     37$
6555  031352  032704  000020                13$:   BIT     #FMTE,R4        ;TEST FORMAT ERROR
6556  031356  001403                          BEQ     14$
6557  031360  104016                          ERROR   16
6558  031362  000137  032372                JMP     37$
6559  031366  032704  004000                14$:   BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
6560  031372  001403                          BEQ     15$
6561  031374  104017                          ERROR   17
6562  031376  000137  032372                JMP     37$
6563  031402  032704  040000                15$:   BIT     #UNS,R4         ;TEST DRIVE UNSAFE
6564  031406  001406                          BEQ     16$
6565  031410  104020                          ERROR   20
6566  031412  052737  000200  005414          BIS     #ABORT,RECODE
6567  031420  000137  032472                JMP     ALLTRM
6568  031424  032704  000002                16$:   BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
6569  031430  001406                          BEQ     17$
6570  031432  104021                          ERROR   21
6571  031434  052737  004000  005414          BIS     #RCLREQ,RECODE
6572  031442  000137  032372                JMP     37$
6573  031446  032704  001000                17$:   BIT     #COE,R4         ;TEST CYLINDER OVERFLOW
6574  031452  001406                          BEQ     18$
6575  031454  104022                          ERROR   22
6576  031456  052737  004000  005414          BIS     #RCLREQ,RECODE
6577  031464  000137  032372                JMP     37$
6578  031470  032704  002000                18$:   BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
6579  031474  001406                          BEQ     19$
6580  031476  104023                          ERROR   23
6581  031500  052737  004000  005414          BIS     #RCLREQ,RECODE
6582  031506  000137  032372                JMP     37$
6583  031512  032765  000040  000036  19$:   BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
6584  031520  001406                          BEQ     20$
6585  031522  104024                          ERROR   24
6586  031524  052737  004000  005414          BIS     #RCLREQ,RECODE
6587  031532  000137  032372                JMP     37$
6588  031536  032704  010000                20$:   BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
6589  031542  001406                          BEQ     21$
6590  031544  104025                          ERROR   25
6591  031546  052737  000200  005414          BIS     #ABORT,RECODE
6592  031554  000137  032372                JMP     37$
6593  031560  032765  100000  000020  21$:   BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
```

```
6594  031566  001403                        BEQ     22$
6595  031570  104026                        ERROR   26
6596  031572  000137  032372                JMP     37$
6597  031576  032704  020000        22$:    BIT     #OPI,R4         ;TEST IF OPI ERROR
6598  031602  001470                        BEQ     29$
6599  031604  052737  000010  005414        BIS     #OPIERR,RECODE
6600  031612  105737  003120                TSTB    DERCNT          ;TEST IF FIRST ERROR
6601  031616  001402                        BEQ     50$
6602  031620  000137  032372                JMP     37$             ;NO - SKIP REPORT
6603  031624  032737  000400  005414  50$:  BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
6604  031632  001403                        BEQ     26$             ;HAS ALREADY OCCURRED
6605  031634  104054                27$:    ERROR   54
6606  031636  000137  032372                JMP     37$             ;GET OUT OF ERROR REPORT
6607  031642  004737  033600        26$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
6608  031646  004737  033270                JSR     PC,RDHDO        ;GET HEADER OF SECTOR 0
6609  031652  032737  000400  005414        BIT     #LEV2ER,RECODE  ;TEST IF ERROR GETTING HDR
6610  031660  001036                        BNE     28$             ;IF YES-GO MAKE ABBREVIATED REPORT
6611  031662  013702  003036                MOV     RKBAS,R2        ;STORE HEADER 0 INTO REGISTERS
6612  031666  042762  000100  000000        BIC     #IE,RKCS1(R2)   ;RESET INTERRUPT ENABLE
6613  031674  016237  000024  001202        MOV     RKDB(R2),$REG10 ;FOR REPORTING
6614  031702  016237  000024  001204        MOV     RKDB(R2),$REG11
6615  031710  016237  000024  001206        MOV     RKDB(R2),$REG12
6616  031716  032762  100000  000000        BIT     #CERR,RKCS1(R2) ;TEST IF ERROR DURING STORAGE
6617  031724  001343                        BNE     27$
6618  031726  105737  003133                TSTB    TSTTYP          ;SEE IF READING WITH OFFSET
6619  031732  001403                        BEQ     54$             ;BR IF NOT
6620  031734  105037  003125                CLRB    ERRCNT          ;CLEAR ERROR COUNT
6621  031740  000401                        BR      55$             ;CONTINUE
6622  031742  104030                54$:    ERROR   30              ;MAKE OPI REPORT
6623  031744  052737  004000  005414  55$:  BIS     #RCLREQ,RECODE  ;SET RECALIBRATE REQUEST BIT
6624  031752  000137  032372                JMP     37$
6625  031756  104054                28$:    ERROR   54
6626  031760  000137  030660                JMP     ER2ENT          ;GO MAKE 2ND LEVEL REPORT
6627  031764  032704  000400        29$:    BIT     #HVRC,R4        ;TEST IF HVRC ERROR
6628  031770  001457                        BEQ     23$
6629  031772  052737  000004  005414        BIS     #HVRCER,RECODE
6630  032000  105737  003120                TSTB    DERCNT          ;TEST IF FIRST ERROR
6631  032004  001402                        BEQ     30$             ;YES - REPORT
6632  032006  000137  032372                JMP     37$             ;JUMP TO RETURN
6633  032012  032737  000400  005414  30$:  BIT     #LEV2ER,RECODE  ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
6634  032020  001403                        BEQ     31$             ;OCCURRED. NO-SKIP EXIT
6635  032022  104055                51$:    ERROR   55
6636  032024  000137  032372                JMP     37$             ;GET OUT OF ERROR REPORT
6637  032030  004737  033600        31$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
6638  032034  004737  033270                JSR     PC,RDHDO        ;GO GET HDR 0
6639  032040  032737  000400  005414        BIT     #LEV2ER,RECODE  ;TEST IF ERROR IN GETTING HDR
6640  032046  001025                        BNE     32$             ;IF YES-GO MAKE ABREVIATED REPORT
6641  032050  013702  003036                MOV     RKBAS,R2        ;GET RK611 BASE ADDRESS
6642  032054  042762  000100  000000        BIC     #IE,RKCS1(R2)   ;CLEAR INTERRUPT ENABLE
6643  032062  016237  000024  001202        MOV     RKDB(R2),$REG10 ;STORE OFF HEADER
6644  032070  016237  000024  001204        MOV     RKDB(R2),$REG11
6645  032076  016237  000024  001206        MOV     RKDB(R2),$REG12
6646  032104  032762  100000  000000        BIT     #CERR,RKCS1(R2) ;TEST IN ANY ERROR IN UNLOAD
6647  032112  001343                        BNE     51$             ;IY YES - GO MAKE SHORT REPORT
6648  032114  104031                        ERROR   31              ;MAKE FULL REPORT
6649  032116  000137  032372                JMP     37$
```

```
6650  032122  104055                    32$:  ERROR  55                    ;ABBREVIATED HVRC ERROR RPORT
6651  032124  000137  030660                  JMP    ER2ENT                 ;GO REPORT 2ND LEVEL ERROR
6652  032130  032704  000200              23$:  BIT    #BSE,R4                ;TEST FOR BAD SECTOR ERROR
6653  032134  001436                          BEQ    33$                    ;NO - SKIP
6654  032136  052737  000002  005414          BIS    #BSERR,RECODE          ;SET ERROR FLAG
6655  032144  016500  000030                  MOV    P.DCYL(R5),R0          ;GET CYL NO.
6656  032150  116501  000027                  MOVB   P.DTS+1(R5),R1         ;GET TRACK NO.
6657  032154  042701  177774                  BIC    #177774,R1             ;CLEAR ALL BITS EXCEPT TRACK
6658  032160  016502  000026                  MOV    P.DTS(R5),R2           ;GET SECTOR IN ERROR
6659  032164  042702  177740                  BIC    #177740,R2             ;CLEAR ALL BITS EXCEPT SECTOR
6660  032170  004737  027176                  JSR    PC,BDSRCK              ;GO SEE IF THIS SECTOR LISTED
6661  032174  032737  001000  005414          BIT    #BADSEC,RECODE         ;TEST RESULT
6662  032202  001073                          BNE    37$                    ;YES - EXIT, NO ERROR OR REPORT
6663  032204  105737  003133                  TSTB   TSTTYP                 ;SEE IF READING WITH OFFSET
6664  032210  001403                          BEQ    56$                    ;BR IF NOT
6665  032212  105037  003125                  CLRB   ERRCNT                 ;CLEAR ERROR COUNT
6666  032216  000465                          BR     37$                    ;CONTINUE
6667  032220  104104                    56$:  ERROR  104                    ;ELSE REPORT BAD BSE
6668  032222  042737  000002  005414          BIC    #BSERR,RECODE          ;RESET BSE ERROR FLAG
6669  032230  000460                          BR     37$                    ;GO EXIT
6670  032232  032704  100000              33$:  BIT    #DCK,R4                ;TEST IF DATA CHECK
6671  032236  001435                          BEQ    36$
6672  032240  052737  000020  005414          BIS    #DCKERR,RECODE         ;SET DATA CHECK ERROR IN RECOVERY CODE
6673  032246  032704  000100                  BIT    #ECH,R4                ;TEST IF ECC IS HARD. IF
6674  032252  001406                          BEQ    34$                    ;YES SET UNCORRECTABLE IN
6675  032254  052737  000040  005414          BIS    #ECCNC,RECODE          ;RECOVERY FLAG AND A 0 IN
6676  032262  005037  001206                  CLR    $REG12                 ;REG12 TO INDICATE UNCORRECTABLE,
6677  032266  000403                          BR     35$                    ;A 1 IN REG 12 FOR CORRECTABLE
6678  032270  012737  000001  001206  34$:    MOV    #1,$REG12
6679  032276  105737  003120              35$:  TSTB   DERCNT                 ;TEST IF FIRST ERROR
6680  032302  001033                          BNE    37$                    ;NO SKIP REPORT
6681  032304  004737  033444                  JSR    PC,GTPKAD              ;GO GET PACK ADDRESS OF ERROR
6682  032310  016537  000060  001202          MOV    P.EPOS(R5),$REG10 ;STORE ECC POSITION &
6683  032316  016537  000062  001204          MOV    P.EPAT(R5),$REG11 ;PATTERN
6684  032324  104032                          ERROR  32                     ;REPORT DCK ERROR
6685  032326  000137  032372                  JMP    37$
6686  032332  032765  040000  000020  36$:    BIT    #WCE,P.CS2(R5)  ;TEST WRITE CHECK ERROR
6687  032340  001414                          BEQ    37$
6688  032342  042737  000200  005414          BIC    #ABORT,RECODE   ;CLEAR ABORT & SET WRITE
6689  032350  052737  000100  005414          BIS    #WCERR,RECODE   ;CHECK ERROR IN RECODE
6690  032356  105737  003120                  TSTB   DERCNT                 ;TEST IF FIRST ERROR
6691  032362  001003                          BNE    37$                    ;NO - SKIP
6692  032364  004737  033444                  JSR    PC,GTPKAD              ;GO GET ADDRESS OF ERROR
6693  032370  104033                          ERROR  33                     ;REPORT WCE
6694
6695  032372  032765  000020  000014  37$:    BIT    #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
6696  032400  001404                          BEQ    43$
6697  032402  104036                          ERROR  36
6698  032404  052737  000200  005414          BIS    #ABORT,RECODE
6699
6700  032412  032765  000040  000014  43$:    BIT    #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
6701  032420  001404                          BEQ    44$
6702  032422  104037                          ERROR  37
6703  032424  052737  000200  005414          BIS    #ABORT,RECODE
6704
6705  032432  032765  004000  000014  44$:    BIT    #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
```

```
6706  032440  001404                        BEQ     46$
6707  032442  104040                        ERROR   40
6708  032444  052737  000200  005414        BIS     #ABORT,RECODE
6709
6710  032452  032765  000010  000014  46$:  BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
6711  032460  001404                        BEQ     ALLTRM
6712  032462  104042                        ERROR   42
6713  032464  052737  000200  005414        BIS     #ABORT,RECODE
6714
6715
6716                                  ;ALL ERRORS MUST EXIT THROUGH THIS POINT
6717
6718  032472  012705  002630        ALLTRM: MOV     #PARMO,R5       ;RESTORE PARAMETER BLOCK SELECTION
6719  032476  012737  020204  003046        MOV     #DCKHDL,A.ABNL  ;SET READ ERROR HANDLER
6720  032504  105737  003133                TSTB    TSTTYP          ;SEE IF READING WITH OFFSETS
6721  032510  001003                        BNE     20$             ;BR IF YES
6722  032512  012737  030640  003046        MOV     #ERRHDL,A.ABNL  ;RESTORE INTERRUPT VECTORS FOR RETRY
6723  032520  012737  027426  003044  20$:  MOV     #ERRFRE,A.NORM  ;DRIVER OPERATIONS, IF ANY
6724  032526  032737  000200  005414        BIT     #ABORT,RECODE   ;IF ABORT IS NOT SET AND
6725  032534  001057                        BNE     48$             ;THE DRIVE IS READY (HAS NOT
6726                                                                ;CYCLED DOWN)
6727  032536  013702  003036                MOV     RKBAS,R2        ;GET BASE ADDRESS
6728  032542  032762  000200  000012        BIT     #RDY,RKDS(R2)   ;TEST IF DRIVE READY SET
6729  032550  001004                        BNE     47$             ;RECALIBRATE REQUIRED BIT IS SET
6730  032552  052737  000200  005414        BIS     #ABORT,RECODE   ;ELSE ABORT WITH ABORT MESSAGE
6731  032560  000445                        BR      48$
6732  032562  032737  004000  005414  47$:  BIT     #RCLREQ,RECODE  ;IF RECALIBRATE IS REQUIRED
6733  032570  001451                        BEQ     BGNRTY          ;FOR RETRY SET UP PARAM
6734  032572  013737  002630  002714        MOV     PARMO,PARM1     ;SAVE COMMAND
6735  032600  112765  000113  000001        MOVB    #RECAL,P.CMND(R5) ;BLOCK TO DO IT.
6736  032606  012737  033072  003046        MOV     #RETANL,A.ABNL
6737  032614  004737  027302                JSR     PC,DRVCAL
6738  032620  013737  002714  002630        MOV     PARM1,PARMO     ;RESTORE COMMAND
6739  032626  012737  020204  003046        MOV     #DCKHDL,A.ABNL  ;SET READ ERROR HANDLER
6740  032634  105737  003133                TSTB    TSTTYP          ;SEE IF READING WITH OFFSETS
6741  032640  001003                        BNE     50$             ;BR IF YES
6742  032642  012737  030640  003046        MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR RETURN
6743  032650  032737  000400  005414  50$:  BIT     #LEV2ER,RECODE  ;IF AN ERROR OCCURRED IN THE
6744  032656  001416                        BEQ     BGNRTY          ;RECAL ATTEMP SET ABORT,
6745  032660  052737  000200  005414        BIS     #ABORT,RECODE   ;PRINT THE RECAL ERROR MESSAGE, AND
6746  032666  104060                        ERROR   60              ;GO REPORT DETAILS
6747  032670  000137  030660                JMP     ER2ENT
6748  032674  104061                48$:    ERROR   61              ;REPORT ABORT-RETRY FAILED
6749
6750                                  ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
6751                                  ;IF THE DRIVE READY BIT IS RESET.
6752
6753  032676  000005                HLTPRG: RESET                   ;DISABLE ALL DEVICES
6754  032700  000000                        HALT                    ;HALT THE CPU
6755  032702  105037  003125                CLRB    ERRCNT          ;CLEAR ERROR COUNT
6756  032706  000005                        RESET                   ;RESET ALL DEVICES
6757  032710  000137  012006                JMP     CMSTRT
6758
6759
6760                                  ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
6761                                  ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
```

```
                                        ; THE INITIATING ROUTINE FOR RETRY.  ANY OTHER ERROR IS TO BE
                                        ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
                                        ;FLAG IS SET AND PROGRAM HALTS.

032714 032737 000136 005414  BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
032722 001404                      BEQ     3$
032724 052737 100000 005414  9$:    BIS     #ANYDER,RECODE
032732 000453                      BR      2$
032734                       3$:
032734 105737 003140               TSTB    NORTRY          ;SEE IF "NO-RETRY" FLAG SET
032740 001371                      BNE     9$              ;BR IF YES
032742 032737 001000 005414        BIT     #BADSEC,RECODE  ;TEST IF BAD SECTOR FLAG SET
032750 001044                      BNE     2$              ;IF YES-EXIT TO CALLER
032752 123737 003125 003126        CMPB    ERRCNT,ERRLMT   ;BEGIN RETRY IF ERROR COUNT HAS
032760 001012                      BNE     1$              ;NOT BEEN EXCEEDED
032762 005037 001174               CLR     $REGS
032766 113737 003125 001174        MOVB    ERRCNT,$REGS    ;GET ERROR RETRY COUNT
032774 104102                      ERROR   102             ;REPORT RETRY UNSUCCESSFUL
032776 052737 000200 005414        BIS     #ABORT,RECODE   ;SET ABORT & QUIT
033004 000734                      BR      HLTPRG
033006 013702 003036         1$:   MOV     RKBAS,R2        ;GET RK BASE ADDRESS
033012 112765 000177 000001        MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
033020 004737 027302               JSR     PC,DRVCAL       ;GO DO IT
033024 012700 005166               MOV     #COMSTR,R0      ;GO AND REESTABLISH THE COMMAND
033030 012025                      MOV     (R0)+,(R5)+     ;AS IT WAS ENTERED INTO THE
033032 012025                      MOV     (R0)+,(R5)+     ;PARAMETER BLOCK
033034 012025                      MOV     (R0)+,(R5)+
033036 012025                      MOV     (R0)+,(R5)+
033040 012025                      MOV     (R0)+,(R5)+
033042 012025                      MOV     (R0)+,(R5)+
033044 012705 002630               MOV     #PARM0,R5
033050 012737 000000 177776        MOV     #PRO,@#PSW      ;LOWER PRIORITY TO ALLOW INTERRUPT
033056 004737 027302               JSR     PC,DRVCAL       ;CALL DRIVER
033062 105037 003125         2$:   CLRB    ERRCNT          ;CLEAR ERROR COUNT
033066 104410                      RESREG                  ;IF RETURN GETS HERE, NO ERROR
033070 000207                      RTS     PC              ;OCCURRED, RECOVERY WAS SUCCESSFUL

033072 152737 000377 003122  RETANL: BISB    #377,DONE       ;SET DONE
033100 052737 000400 005414        BIS     #LEV2ER,RECODE  ;SET LEVEL TWO ERROR
033106 000207                      RTS     PC
033110 152737 000377 003122  RETNML: BISB    #377,DONE       ;SET DONE
033116 000207                      RTS     PC

                                        ;;*****************************************************************
                                        .SBTTL  TIME OUT PROCESSOR ROUTINE
                                        ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
                                        ;*GATHERING DUTIES.
                                        ;;*****************************************************************
033120                          TOPROC:
033120 104407                      SAVREG
033122 013702 003036               MOV     RKBAS,R2
033126 012701 001174               MOV     #$REGS,R1       ;SET UP R1 FOR RK REGISTER STORAGE
033132 016221 000000               MOV     RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
033136 016221 000010               MOV     RKCS2(R2),(R1)+ ;REGISTERS
```

```
 6818  033142  016221  000020                    MOV     RKDC(R2),(R1)+
 6819  033146  016221  000006                    MOV     RKDA(R2),(R1)+
 6820  033152  016221  000002                    MOV     RKWC(R2),(R1)+
 6821  033156  016221  000004                    MOV     RKBA(R2),(R1)+
 6822  033162  016221  000016                    MOV     RKASOF(R2),(R1)+
 6823  033166  016221  000012                    MOV     RKDS(R2),(R1)+
 6824  033172  016221  000014                    MOV     RKER(R2),(R1)+
 6825  033176  005000                            CLR     R0              ;THIS CODE WILL ATTEMPT TO
 6826                                                                    ;RETRIEVE THE STATUS FROM THE
 6827                                                                    ;DRIVE.
 6828  033200  012705  002714                    MOV     #PARM1,R5       ;SET UP TO USE PARM1
 6829  033204  112765  000141  000001            MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
 6830  033212  004737  027302                    JSR     PC,DRVCAL       ;CALL DRIVER
 6831  033216  032765  000100  000014            BIT     #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
 6832  033224  001403                            BEQ     1$              ;NO - SKIP
 6833  033226  052737  002000  005414            BIS     #TWOTOS,RECODE  ;SET TWO TIMEOUTS FLAG
 6834  033234  062705  000040              1$:   ADD     #P.ADD,R5       ;BUMP R5 TO POINT TO DRIVE STATUS
 6835  033240  012521                            MOV     (R5)+,(R1)+     ;MOVE ALL THE DRIVE STATUS INTO THE
 6836  033242  012521                            MOV     (R5)+,(R1)+     ;TEMP REGS FOR REPORTING.
 6837  033244  012521                            MOV     (R5)+,(R1)+
 6838  033246  012521                            MOV     (R5)+,(R1)+
 6839  033250  012521                            MOV     (R5)+,(R1)+
 6840  033252  012521                            MOV     (R5)+,(R1)+
 6841  033254  012521                            MOV     (R5)+,(R1)+
 6842  033256  012521                            MOV     (R5)+,(R1)+
 6843
 6844  033260  012705  002630                    MOV     #PARM0,R5       ;RESTORE PARM 0
 6845  033264  104410                            RESREG
 6846  033266  000207                            RTS     PC
 6847
 6848
 6849
 6850                                    ;;*****************************************************************
 6851                                    .SBTTL  READ HEADER 0 ROUTINE
 6852                                    ;;*****************************************************************
 6853  033270                            RDHD0:
 6854  033270  104407                            SAVREG
 6855  033272  016501  000026                    MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
 6856  033276  016500  000052                    MOV     P.B10(R5),R0    ;GET THE CYLINDER ADRS
 6857  033302  042700  160017                    BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
 6858  033306  006200                            ASR     R0              ;OFF UNUSED BITS AND POSITION
 6859  033310  006200                            ASR     R0              ;FOR USE AS THE DESIRED
 6860  033312  006200                            ASR     R0              ;CYLINDER IN THE READ
 6861  033314  006200                            ASR     R0              ;HEADER COMMAND.
 6862  033316  012705  002714                    MOV     #PARM1,R5       ;SET UP TO USE P.B. 1
 6863  033322  010165  000004                    MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
 6864  033326  010065  000002                    MOV     R0,P.CYLN(R5)   ;SET CYL NO.
 6865  033332  112765  000177  000001            MOVB    #SUBCLR,P.CMND(R5)      ;SET S.S. CLEAR CMND
 6866  033340  012737  000000  177776            MOV     #PR0,@#PSW      ;ALLOW INTERRUPTS
 6867  033346  004737  027302                    JSR     PC,DRVCAL       ;INIT THE S.S.
 6868  033352  112765  000125  000001            MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
 6869  033360  004737  027302                    JSR     PC,DRVCAL       ;DO A READ HEADER
 6870  033364  012762  000025  000000            MOV     #25,RKCS1(R2)   ;DO A READ HDR, WITH FMT BIT = 0
 6871  033372  032762  000200  000000  24$:     BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
 6872  033400  001774                            BEQ     24$
 6873  033402  012762  010025  000000            MOV     #10025,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
```

```
6874  033410  032762  000200  000000  26$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
6875  033416  001774                          BEQ     26$
6876  033420  142765  000020  000007          BICB    #8.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
6877  033426  153765  003124  000007          BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
6878  033434  012705  002630                  MOV     #PARM0,R5       ;RESTORE P.B. 0 ADDRESS
6879  033440  104410                          RESREG                  ;RESTORE R0-R5
6880  033442  000207                          RTS     PC              ;RETURN
6881
6882
6883
6884                                  ;;*****************************************************************
6885                                  .SBTTL  GET PACK ADDRESS ROUTINE
6886                                  ;;*****************************************************************
6887  033444  016537  000030  001174  GTPKAD: MOV     P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
6888  033452  005037  001176                  CLR     $REG6           ;CLEAR REGISTERS FOR
6889  033456  005037  001200                  CLR     $REG7           ;TRACK & SECTOR STORAGE
6890  033462  116537  000026  001200          MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
6891  033470  116537  000027  001176          MOVB    P.DTS+1(R5),$REG6
6892  033476  032737  000004  005414          BIT     #HVRCER,RECODE  ;SEE IF HVRC ERROR
6893  033504  001034                          BNE     5$              ;BR IF YES
6894  033506  005737  001200                  TST     $REG7             ;ADJUST THE ADDRESS CONTAINED IN
6895                                                                  ;THE RK REGISTERS FOR THE AUTOMATIC
6896  033512  001403                          BEQ     1$              ;INCREMENT
6897  033514  005337  001200                  DEC     $REG7
6898  033520  000426                          BR      5$
6899  033522  032765  010000  000016  1$:     BIT     #CFMT,P.CS1(R5)
6900  033530  001404                          BEQ     2$
6901  033532  012737  000023  001200          MOV     #19.,$REG7
6902  033540  000403                          BR      3$
6903  033542  012737  000025  001200  2$:     MOV     #21.,$REG7
6904  033550  005737  001176          3$:     TST     $REG6
6905  033554  001403                          BEQ     4$
6906  033556  005337  001176                  DEC     $REG6
6907  033562  000405                          BR      5$
6908  033564  012737  000002  001176  4$:     MOV     #2,$REG6
6909  033572  005337  001174                  DEC     $REG5
6910  033576  000207                  5$:     RTS     PC
6911
6912
6913
6914                                  ;;*****************************************************************
6915                                  .SBTTL  BUILD EXPECTED HEADER
6916                                  ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
6917                                  ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
6918                                  ;*7 FOR REPORTING.
6919                                  ;;*****************************************************************
6920  033600  104407                  BLDEXH: SAVREG
6921  033602  016537  000030  001174          MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
6922  033610  016501  000026                  MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
6923  033614  042701  174377                  BIC     #174377,R1      ;CLEAR ALL BUT TRACK BITS
6924  033620  006201                          ASR     R1              ;AND SECTOR. SHIFT THE TRACK
6925  033622  006201                          ASR     R1              ;OVER TO CONFORM TO HEADER FORMAT
6926  033624  006201                          ASR     R1              ;CHECK THE FORMAT BIT AND
6927                                                                  ;IF SET, SET THE HEADER FORMAT
6928  033626  016537  000026  001176          MOV     P.DTS(R5),$REG6 ;BIT.
6929  033634  042737  177740  001176          BIC     #177740,$REG6   ;CLEAR ALL BUT SECTOR
```

```
 6930  033642  060137  001176                    ADD     R1,$REG6        ;ADD TRACK AND SECTOR TOGETHER
 6931  033646  052737  140000  001176            BIS     #140000,$REG6   ;INSERT BSE BITS
 6932  033654  032765  010000  000016            BIT     #CFMT,P.CS1(R5)
 6933  033662  001403                            BEQ     23$
 6934  033664  052737  001000  001176            BIS     #1000,$REG6
 6935  033672  013737  001174  001200  23$:      MOV     $REG5,$REG7     ;COMPUTE THE HEADER VRC
 6936  033700  013701  001176                    MOV     $REG6,R1
 6937  033704  043737  001176  001200            BIC     $REG6,$REG7
 6938  033712  043701  001174                    BIC     $REG5,R1
 6939  033716  050137  001200                    BIS     R1,$REG7
 6940  033722  104410                            RESREG
 6941  033724  000207                            RTS     PC
 6942
```

```
                                .SBTTL  RK611/RK06 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.08)

                                ;*COPYRIGHT (C) 1975
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MA. 01754
                                ;*AUTHOR: ROY SPITZER

                                .SBTTL   *WATCH-DOG TIMER

                                ;;*********************************************************************
                                ;*
                                ;*      THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06 UNIBUS
                                ;*      SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
                                ;*      REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
                                ;*      THE RK06 DRIVER WILL USE THE LOCATION W.MTIM FOR
                                ;*      MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
                                ;*      WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
                                ;*      DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
                                ;*      (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
                                ;*      IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
                                ;*      TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
                                ;*      REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
                                ;*
                                ;*      THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
                                ;*      OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
                                ;*      THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
                                ;*      LIMIT FOR ALL OTHER COMMANDS.
                                ;*
                                ;*      FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
                                ;*      WATCH UP TO 8 OPERATIONS SIMULATEOUSLY. FOR SEQUENTIAL
                                ;*      OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
                                ;*
                                ;*CALL   JSR     PC,W.WTCH
                                ;*      RETURN          IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
                                ;*
                                ;*      OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
                                ;*      BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
                                ;*      IN THE PROGRAM DEVICE STATUS REGISTER OF THE
                                ;*      APPROPRIATE PARAMETER BLOCK WILL BE SET.
                                ;*
                                ;;*********************************************************************

033726  010546          W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
033730  010446                  MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
033732  010346                  MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
033734  010246                  MOV     R2,-(SP)        ;SAVE R2 ON STACK
033736  013746  177776          MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
033742  005337  003056          DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
033746  001034                  BNE     20$             ;IF NOT ZERO RETURN
033750  013737  003060  003056  MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
033756  105737  003100          TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
033762  001426                  BEQ     20$             ;NO, RETURN
033764  013737  003042  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
033772  013702  003036          MOV     RKBAS,R2        ;LOAD BASE OF RK06 REGISTERS
033776  005337  003114          DEC     W.DRV           ;DECREMENT COMMAND TIMER
034002  001016                  BNE     20$             ;RETURN IF NO TIME OUT
```

```
 6999  034004  105037  003100                  CLRB    W.TIME          ;RESET TIMING INDICATOR
 7000  034010  013705  003112                  MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
 7001                                                                  ;  TABLE FOR INDEXING
 7002  034014  052765  000100  000014          BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
 7003  034022  020537  003054                  CMP     R5,O.WAIT       ;CHECK IF DRIVER IS WAITING FOR
 7004                                                                  ;  COMMAND COMPLETION
 7005  034026  001002                          BNE     5$              ;NO, DO NOT ALTER WAITING FOR
 7006                                                                  ;  COMMAND COMPLETION
 7007  034030  005037  003054                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 7008  034034  004737  037310          5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
 7009  034040  012637  177776          20$:    MOV     (SP)+,PS        ;RESTORE PSW
 7010  034044  012602                          MOV     (SP)+,R2        ;RESTORE R2
 7011  034046  012603                          MOV     (SP)+,R3        ;RESTORE R3
 7012  034050  012604                          MOV     (SP)+,R4        ;RESTORE R4
 7013  034052  012605                          MOV     (SP)+,R5        ;RESTORE R5
 7014  034054  000207                          RTS     PC              ;RETURN
```

```
                        .SBTTL   *RK06 INTERRUPT SERVICE ROUTINE

                        ;;*****************************************************************
                        ;;*
                        ;;*      THIS ROUTINE WILL SERVICE ALL RK06 INTERRUPTS.
                        ;;*
                        ;;*      UPON RECEIVING AN INTERRUPT, THIS ROUTINE WILL
                        ;;*      PERFORM ONE OF THE FOLLOWING SERVICES:
                        ;;*
                        ;;*      1.) SERVICE PORT WAS SEIZED BY OTHER PORT
                        ;;*      2.) SERVICE DRIVER IS WAIT FOR COMMAND COMPLETION
                        ;;*      3.) SERVICE POSITIONING COMPLETION
                        ;;*      4.) REQUEUE COMMAND IF DRIVE WAS RELEASED
                        ;;*          FOR THE QUEUED RK06 DRIVER.
                        ;;*      5.) IF NO SERVICE IS REQUIRED, THE COMMAND WILL BE ISSUED
                        ;;*          FOR THE QUEUED RK06 DRIVER.
                        ;;*
                        ;;*      THREE LINKS ARE PROVIDED TO THE DRIVING PROGRAM.
                        ;;*      THEY ARE:
                        ;;*
                        ;;*      1.) A.NORM ADDRESS OF NORMAL RETURN (SUCESSFUL COMPLETION OF COMMAND)
                        ;;*      2.) A.ABNL ADDRESS OF ABNORMAL RETURN (UNSUCESSFUL COMPLETION OF COMMAND)
                        ;;*      3.) A.CONT ADDRESS OF CONTROL ERROR RETURN
                        ;;*
                        ;;*      FOR NORMAL AND ABNORMAL RETURNS, THE ADDRESS OF THE APPROPRIATE
                        ;;*      PARAMETER BLOCK WILL BE IN R5.
                        ;;*
                        ;;*      FOR THE CONTROLLER ERROR RETURN, THE LOCATION E.CONT CONTAINS
                        ;;*      THE REASON FOR THE CONTROLLER ERROR.
                        ;;*
                        ;;*      ROUTINES USED:
                        ;;*              C.OPT   (QUEUED ONLY)
                        ;;*              Q.PUSH  (QUEUED ONLY)
                        ;;*              Q.RMOV  (QUEUED ONLY)
                        ;;*              R.CONT  (SEQUENTIAL ONLY)
                        ;;*              R.NORM  (SEQUENTIAL ONLY)
                        ;;*              R.ABNL  (SEQUENTIAL ONLY)
                        ;;*              I.CSTS
                        ;;*              I.STAT
                        ;;*              I.ISSU
                        ;;*              I.CCLR
                        ;;*
                        ;;*****************************************************************

034056  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
034060  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
034062  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
034064  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
034066  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
034070  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
034072  013702  003036          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
034076  016237  000010  003002  MOV     RKCS2(R2),T.CS2 ;STORE CS2
034104  032737  001000  003002  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
034112  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
034114  052737  100000  003052  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
034122  004737  037334          JSR     PC,R.CONT       ;REPORT ERROR
```

```
7071  034126  000137  036306                    JMP     I.RTRN          ;RETURN
7072
7073  034132  105737  003074          1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
7074  034136  001410                             BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
7075  034140  100403                             BMI     5$              ;CHECK IF RELEASE COMMAND
7076  034142  105037  003074                    CLRB    I.ISRL          ;YES,CLEAR FLAG
7077  034146  000473                             BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
7078
7079  034150  105037  003074          5$:       CLRB    I.ISRL          ;CLEAR FLAG
7080  034154  000137  035270                    JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
7081
7082  034160  032737  010400  003002  6$:       BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
7083                                                                    ;  UNIT FIELD ERROR
7084  034166  001413                             BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
7085  034170  013704  003002                    MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
7086  034174  042704  177770                    BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
7087  034200  013705  003112                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
7088  034204  016237  000000  003000            MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
7089  034212  000137  034500                    JMP     I.ERRC          ;REPORT ERROR
7090
7091  034216  016237  000012  003020  7$:       MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
7092  034224  032737  000001  003020            BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
7093                                                                    ;  PORT
7094  034232  001041                             BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
7095
7096                                            ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
7097  034234  032737  164000  003002            BIT     #DLT!WCE!UPE!NEM,T.CS2
7098
7099  034242  001007                             BNE     10$             ;INDICATE ERROR
7100  034244  016237  000014  003016            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
7101
7102                                  ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
7103  034252  032737  125700  003016            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
7104
7105  034260  001407                             BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
7106
7107  034262  052737  000010  003052  10$:      BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
7108  034270  004737  037334                    JSR     PC,R.CONT       ;REPORT ERROR
7109  034274  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7110
7111  034300  105037  003100          11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
7112  034304  005037  003114                    CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
7113  034310  013705  003112                    MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
7114                                                                    ;  ADDRESS
7115  034314  052765  010000  000014            BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
7116                                                                    ;  PROGRAM DRIVE STATUS REGISTER
7117  034322  005037  003054                    CLR     0.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
7118  034326  004737  037310                    JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
7119  034332  000137  036306                    JMP     I.RTRN          ;GO RESTORE REGISTERS
7120
7121  034336  013705  003054          I.IOO:    MOV     0.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
7122  034342  001002                             BNE     2$              ;IS COMMAND WAITING PROCESSING
7123                                                                    ;  YES, DO PROCESSING
7124  034344  000137  035270                    JMP     I.ATTN          ;NO, PROCESS ATTENTION
7125
7126  034350  013704  003002          2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
 7127  034354  042704  177770                    BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
 7128
 7129
 7130  034360  126504  000000                    CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
 7131  034364  001401                            BEQ     3$              ;YES, CONTINUE
 7132  034366  000000                            HALT                    ;NO, DRIVER ERROR
 7133  034370  122765  000164  000001  3$:       CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 7134  034376  001002                            BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
 7135  034400  000137  034736                    JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
 7136
 7137  034404  005037  003054          10$:      CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 7138  034410  005037  003114                    CLR     W.DRV           ;CLEAR WATCH-DOG TIME
 7139  034414  105037  003100                    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 7140  034420  016237  000000  003000            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
 7141  034426  032737  100000  003000            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
 7142  034434  001021                            BNE     I.ERRC          ;YES, PROCESS ERROR
 7143  034436  016237  000016  003014            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 7144  034444  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
 7145  034452  001004                            BNE     15$             ;YES, REPORT ERROR
 7146  034454  004737  037322                    JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
 7147  034460  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
 7148
 7149  034464  052765  000010  000014  15$:      BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
 7150
 7151  034472  004737  036756          I.ERRA:   JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
 7152  034476  000405                            BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
 7153
 7154  034500  013765  003000  000016  I.ERRC:   MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
 7155  034506  004737  037000                    JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
 7156  034512  016265  000032  000062  I.ERR:    MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
 7157  034520  016265  000030  000060            MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
 7158  034526  004037  036324                    JSR     R0,I.CCLR       ;CLEAR CONTROLLER
 7159  034532  036306                            I.RTRN                  ;ERROR RETURN
 7160  034534  032765  010400  000020            BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
 7161                                                                    ;  UNIT FIELD ERROR
 7162  034542  001046                            BNE     5$              ;YES, REPORT ERROR
 7163  034544  004037  037062                    JSR     R0,I.STAT       ;GATHER DRIVE STATUS
 7164  034550  036306                            I.RTRN                  ;ERROR RETURN
 7165  034552  112737  000005  003000            MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
 7166  034560  004037  036406                    JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
 7167  034564  036306                            I.RTRN                  ;ERROR RETURN
 7168  034566  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
 7169  034574  001407                            BEQ     2$              ;NO, INDICATE DRIVE ERROR
 7170  034576  052737  000020  003052            BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
 7171                                                                    ;  WITH CLEAR
 7172  034604  004737  037334                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
 7173  034610  000137  036306                    JMP     I.RTRN          ;GO RESTORE REGISTERS
 7174
 7175  034614  032737  040000  003024  2$:       BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
 7176  034622  001403                            BEQ     3$              ;YES, CHECK FAULT
 7177  034624  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
 7178  034632  032737  001000  003026  3$:       BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
 7179  034640  001407                            BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
 7180  034642  052737  002000  003052            BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
 7181  034650  004737  037334                    JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
 7182  034654  000137  036306                    JMP     I.RTRN          ;RETURN
```

```
7183
7184  034660  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
7185  034666  001017                          BNE     10$             ;YES, GO REPORT ERROR
7186  034670  032737  020000  003024          BIT     #S.PIP,T.MR2    ;CHECK IF DRIVE IS CYCLING DOWN
7187  034676  001413                          BEQ     10$             ;NO, REPORT ERROR
7188  034700  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
7189  034706  113737  003101  003100          MOVB    INTMSK,W.TIME   ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
7190  034714  013737  003064  003114          MOV     W.8SEC,W.DRV
7191  034722  000137  036306                  JMP     I.RTRN          ;GO RESTORE REGISTERS
7192
7193  034726  004737  037310          10$:    JSR     PC,R.ABNL       ;GO REPORT ERROR
7194  034732  000137  036306                  JMP     I.RTRN          ;GO RESTORE REGISTERS
7195
7196                                  .SBTTL   *READ ALL HEADERS INTERRUPT SEQUENCE
7197
7198  034736  016237  000000  003000  I.HDAL: MOV     RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
7199                                                                  ;  ERROR
7200  034744  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
7201  034752  001422                          BEQ     5$              ;NO, CHECK FOR ATTENTION
7202
7203  034754  005037  003054                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETE
7204  034760  105037  003100                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
7205  034764  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
7206  034770  013765  003000  000016          MOV     T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
7207  034776  004737  037000                  JSR     PC,I.CSTI       ;STORE CONTROLLER REGISTERS
7208  035002  004037  036324                  JSR     R0,I.CCLR       ;CLEAR CONTROLLER
7209  035006  036306                          I.RTRN                  ;ERROR RETURN
7210  035010  004737  037310                  JSR     PC,R.ABNL       ;INDICATE ERROR RETURN
7211  035014  000137  036306                  JMP     I.RTRN          ;RESTORE REGISTERS
7212
7213  035020  016237  000016  003014  5$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7214  035026  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
7215  035034  001410                          BEQ     7$              ;NO, CHECK IF READ ALL HEADERS
7216  035036  005037  003054                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETION
7217  035042  105037  003100                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
7218  035046  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
7219  035052  000137  034472                  JMP     I.ERRA          ;GO REPORT ERROR
7220
7221  035056  013701  003070          7$:     MOV     HDR.AD,R1       ;GET MAIN MEMORY ADDRESS
7222  035062  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET FIRST WORD OF HEADER
7223  035066  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET SECOND WORD OF HEADER
7224  035072  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET THIRD WORD OF HEADER
7225  035076  010137  003070                  MOV     R1,HDR.AD       ;STORE ADDRESS FOR NEXT HEADER
7226  035102  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
7227  035110  032737  100000  003002          BIT     #DLT,T.CS2      ;CHECK FOR DATA LATE
7228  035116  001055                          BNE     35$             ;YES, REPORT ERROR
7229  035120  005337  003072                  DEC     HDR.CT          ;DECREMENT NUMBER OF HEADER YET TO READ
7230  035124  001026                          BNE     25$             ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
7231  035126  005037  003054                  CLR     O.WAIT          ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
7232  035132  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT FOR THIS DRIVE
7233  035136  105037  003100                  CLRB    W.TIME          ;CLEAR WATCH DOG TIME ON THIS DRIVE
7234  035142  012762  000003  000026          MOV     #3,RKMR1(R2)    ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
7235  035150  112737  000001  003000          MOVB    #DR.SEL,T.CS1   ;LOAD SELECT COMMAND
7236  035156  004037  036406                  JSR     R0,I.ISSU       ;GET SECTOR COUNT
7237  035162  036306                          I.RTRN                  ;ERROR RETURN
7238  035164  013765  003026  000056          MOV     T.MR3,P.B11(R5) ;LOAD SECTOR COUNT
```

```
7239  035172  004737  037322                    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
7240  035176  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7241
7242  035202  016562  000002  000020  25$:      MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
7243  035210  016562  000004  000006            MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
7244  035216  116565  000007  000017            MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7245  035224  042765  165777  000016            BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
7246                                                                    ;  DRIVE TYPE
7247  035232  112765  000125  000016            MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
7248  035240  016562  000016  000000            MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
7249  035246  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7250
7251  035252  052737  000400  003052  35$:      BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
7252  035260  004737  037334                    JSR     PC,R.CONT       ;REPORT ERROR
7253  035264  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7254
7255                                    .SBTTL  *DRIVE ATTENTION SCANNER
7256
7257  035270  016237  000000  003000  I.ATTN:   MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
7258                                                                    ;  REGISTER 1 FOR COMPARISON
7259  035276  032737  100000  003000            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
7260  035304  001441                            BEQ     5$              ;NO, CHECK IF ATTENTION
7261
7262                                            ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
7263  035306  032737  164000  003002            BIT     #DLT!WCE!UPE!NEM,T.CS2
7264
7265  035314  001007                            BNE     1$              ;INDICATE ERROR
7266  035316  016237  000014  003016            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
7267
7268                                    ;       CHECK FOR DATA TRANSFER ERROR TYPE
7269  035324  032737  125700  003016            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
7270
7271  035332  001407                            BEQ     2$              ;NO DATA TRANSFER ERROR
7272
7273  035334  052737  000010  003052  1$:       BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
7274  035342  004737  037334                    JSR     PC,R.CONT       ;REPORT ERROR
7275  035346  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7276
7277  035352  013704  003002          2$:       MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
7278  035356  042704  177770                    BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
7279  035362  105037  003100                    CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
7280  035366  005037  003114                    CLR     W.DRV           ;RESET TIMER VALUE
7281  035372  013705  003112                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
7282
7283                                    ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
7284                                    ;         IN PROGRAM DEVICE STATUS REGISTER
7285  035376  042765  000006  000014            BIC     #DRVPOS!DRVPDT,P.PRST(R5)
7286
7287  035404  000137  034500                    JMP     I.ERRC          ;GO REPORT ERROR
7288
7289  035410  032737  040000  003000  5$:       BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
7290  035416  001002                            BNE     6$              ;YES, PROCESS INTERRUPT
7291  035420  000137  036306                    JMP     I.RTRN          ;RESTORE REGISTERS
7292
7293  035424  016237  000016  003014  6$:       MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7294  035432  105737  003015                    TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
7295  035436  001007                        BNE     7$                ;YES GO PROCESS INTERRUPT
7296  035440  052737  000002  003052        BIS     #E.NOAT,E.CONT    ;SET NO ATTENTION IN ATTENTION SUMMARY
7297  035446  004737  037334                JSR     PC,R.CONT         ;GO REPORT ERROR
7298  035452  000137  036306                JMP     I.RTRN            ;GO RESTORE REGISTERS
7299
7300  035456  133737  003101  003015  7$:   BITB    INTMSK,T.ASOF+1   ;CHECK IF DESIRED INTERRUPT
7301  035464  001007                        BNE     8$                ;YES, GO PROCESS IT
7302  035466  052737  000004  003052        BIS     #E.UATT,E.CONT    ;SET UNSOLICATED ATTENTION
7303  035474  004737  037334                JSR     PC,R.CONT         ;GO REPORT ERROR
7304  035500  000137  036306                JMP     I.RTRN            ;GO RESTORE REGISTERS
7305
7306  035504  013705  003112          8$:   MOV     PBLKT,R5          ;STORE PARAMETER BLOCK TABLE
7307  035510  116504  000000                MOVB    P.DRVN(R5),R4     ;STORE DRIVE NUMBER
7308  035514  032765  020000  000014        BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
7309  035522  001402                        BEQ     11$               ;NO, CONTINUE
7310  035524  000137  036206                JMP     I.UNLD            ;SERVICE DRIVE IN POSITION AFTER ERROR
7311
7312  035530  042765  000002  000014  11$:   BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
7313  035536  005062  000026                CLR     RKMR1(R2)         ;CLEAR MAINTENANCE REGISTER 1
7314  035542  112737  000001  003000        MOVB    #DR.SEL,T.CS1     ;LOAD COMMAND
7315  035550  004037  036406                JSR     R0,I.ISSU         ;SELECT DRIVE WITH ATTENTION HIGH
7316  035554  036306                        I.RTRN                    ;ERROR RETURN
7317  035556  013765  003026  000042        MOV     T.MR3,P.B00(R5)   ;STORE STATUS BYTE 00 MESS B
7318  035564  032765  000200  000042        BIT     #S.FLT,P.B00(R5)  ;CHECK IF DRIVE FAULT
7319  035572  001401                        BEQ     12$               ;NO, CHECK FOR DRIVE STATUS CHANGE
7320  035574  000461                        BR      I.AERR            ;PROCESS ERROR
7321
7322  035576  013765  003024  000040  12$:   MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
7323  035604  032765  040000  000040        BIT     #S.DSC,P.A00(R5)  ;CHECK FOR DRIVE STATUS CHANGE
7324  035612  001004                        BNE     13$               ;YES, PROCESS DRIVE STATUS CHANGE
7325  035614  052765  004000  000014        BIS     #NODSC,P.PRST(R5)  ;SET NO DRIVE STATUS CHANGE
7326  035622  000446                        BR      I.AERR            ;PROCESS ERROR
7327
7328  035624  112737  000005  003000  13$:   MOVB    #DR.CLR,T.CS1     ;LOAD COMMAND
7329  035632  004037  036406                JSR     R0,I.ISSU         ;CLEAR DRIVE STATUS CHANGE
7330  035636  036306                        I.RTRN                    ;ERROR RETUN
7331  035640  013765  003014  000032        MOV     T.ASOF,P.ASOF(R5)  ;STORE ATTENTION SUMMARY
7332  035646  133765  003101  000033        BITB    INTMSK,P.ASOF+1(R5)  ;CHECK IF ATTENTION RESET
7333  035654  001407                        BEQ     15$               ;YES, CONTINUE INTERRUPT PROCESSING
7334  035656  052737  000020  003052        BIS     #E.CLAT,E.CONT    ;SET ATTENTION DID NOT RESET
7335                                                                  ;  WITH DRIVE CLEAR
7336  035664  004737  037334                JSR     PC,R.CONT         ;FLAG ERROR
7337  035670  000137  036306                JMP     I.RTRN            ;RESTORE REGISTERS
7338
7339  035674  013765  003024  000040  15$:   MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
7340  035702  032765  040000  000040        BIT     #S.DSC,P.A00(R5)  ;CHECK IF DRIVE STATUS CHANGE
7341                                                                  ;  RESET
7342  035710  001404                        BEQ     16$               ;YES, CONTINUE INTERRUPT PROCESSING
7343  035712  052765  000040  000014        BIS     #DRVDSC,P.PRST(R5)  ;SET DRIVE STATUS CHANGE DID NOT CLEAR
7344  035720  000407                        BR      I.AERR            ;GO PROCESS ERROR
7345
7346  035722  105037  003100          16$:   CLRB    W.TIME            ;RESET TIMING ON THIS DRIVE
7347  035726  005037  003114                CLR     W.DRV             ;CLEAR DRIVE TIMING COUNT
7348  035732  004737  037322                JSR     PC,R.NORM         ;REPORT SUCCESSFUL COMMAND COMPLETETION
7349  035736  000563                        BR      I.RTRN            ;RESTORE REGISTERS
7350
```

```
                                        .SBTTL   *ATTENTION ERROR HANDLER
                                        
035740  042765  000004  000014  I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
                                                                   ;  OF DATA TRANSFER
035746  105037  003100                  CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
035752  005037  003114                  CLR     W.DRV              ;RESET WATCH-DOG TIME
035756  042765  177741  000016          BIC     #177741,P.CS1(R5)  ;KEEP COMMAND ISSUED
035764  042737  000036  003000          BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
035772  053765  003000  000016          BIS     T.CS1,P.CS1(R5)    ;MAKE GOOD MESSAGE
036000  013765  003002  000020          MOV     T.CS2,P.CS2(R5)    ;STORE CONTROLLER REGISTERS
036006  013765  003004  000022          MOV     T.WCR,P.WCR(R5)
036014  013765  003006  000024          MOV     T.BA,P.BAR(R5)
036022  013765  003010  000026          MOV     T.DA,P.DTS(R5)
036030  013765  003012  000030          MOV     T.DC,P.DCYL(R5)
036036  013765  003014  000032          MOV     T.ASOF,P.ASOF(R5)
036044  013765  003016  000034          MOV     T.ER,P.ER(R5)
036052  013765  003020  000036          MOV     T.DS,P.DS(R5)
036060  004037  037062                  JSR     R0,I.STAT          ;GATHER DRIVE STATUS
036064  036306                          I.RTRN                     ;ERROR RETURN
036066  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD COMMAND
036074  004037  036406                  JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
036100  036306                          I.RTRN                     ;ERROR RETURN
036102  133737  003101  003015          BITB    INTMSK,T.ASOF+1    ;CHECK IF ATTENTION RESET
036110  001407                          BEQ     2$                 ;YES, FLAG DRIVE ERROR
036112  052737  000020  003052          BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
036120  004737  037334                  JSR     PC,R.CONT          ;REPORT ERROR
036124  000137  036306                  JMP     I.RTRN             ;RESTORE REGISTERS
                                        
036130  032765  000020  000014  2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
036136  001017                          BNE     10$                ;YES, REPORT ERROR
036140  032737  020000  003024          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
036146  001413                          BEQ     10$                ;NO, REPORT ERROR
036150  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
036156  113737  003101  003100          MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
036164  013737  003064  003114          MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
036172  000137  036306                  JMP     I.RTRN             ;RESTORE REGISTERS
                                        
036176  004737  037310          10$:    JSR     PC,R.ABNL          ;REPORT ERROR
036202  000137  036306                  JMP     I.RTRN             ;RESTORE REGISTERS
                                        
                                        .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD
                                        
036206  052765  020000  000014  I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
036214  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
036222  004037  036406                  JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
036226  036306                          I.RTRN                     ;ERROR RETURN
036230  136437  003101  003015          BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
036236  001406                          BEQ     15$                ;YES, CONTINUE
036240  012737  000020  003052          MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
036246  004737  037334                  JSR     PC,R.CONT          ;REPORT ERROR
036252  000415                          BR      I.RTRN             ;RESTORE REGISTERS
                                        
036254  032737  040000  003024  15$:    BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
036262  001403                          BEQ     20$                ;YES, CONTINUE
036264  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
036272  105037  003100          20$:    CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

```
 7407  036276  005037  003114                CLR     W.DRV           ;CLEAR TIME COUNT
 7408  036302  004737  037310                JSR     PC,R.ABNL       ;REPORT ERROR
 7409
 7410  036306  012600              I.RTRN:   MOV     (SP)+,R0        ;RESTORE R0
 7411  036310  012601                        MOV     (SP)+,R1        ;RESTORE R1
 7412  036312  012602                        MOV     (SP)+,R2        ;RESTORE R2
 7413  036314  012603                        MOV     (SP)+,R3        ;RESTORE R3
 7414  036316  012604                        MOV     (SP)+,R4        ;RESTORE R4
 7415  036320  012605                        MOV     (SP)+,R5        ;RESTORE R5
 7416  036322  000002                        RTI                     ;RETURN
 7417
```

```
                                .SBTTL   *CONTROLLER CLEAR ROUTINE

                                ;;**********************************************************************
                                ;*
                                ;*      THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
                                ;*      AND CHECK IF THE CONTROLLER ERRORS ARE RESET.  IF THE ERROR IS NOT
                                ;*      CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
                                ;*      E.CCLR SET IN E.CONT.
                                ;*
                                ;*      REGISTER        USE
                                ;*      --------        ---
                                ;*
                                ;*      R2              ADDRESS OF RK06 REGISTERS
                                ;*      R5              ADDRESS OF PARAMETER BLOCK
                                ;*
                                ;*CALL  JSR     R0,I.CCLR
                                ;*      <ADDRESS OF ERROR RETURN>
                                ;*      RETURN
                                ;*
                                ;;**********************************************************************

036324  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
036332  016237  000000  003000          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
036340  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
036346  001407                        BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
036350  052737  000001  003052        BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
036356  004737  037334                JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
036362  011000                        MOV     (R0),R0         ;SET UP ERROR RETURN
036364  000200                        RTS     R0              ;RETURN

036366  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
036374  112737  177777  003074        MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
036402  005720                        TST     (R0)+           ;ADJUST FOR NORMAL RETURN
036404  000200                        RTS     R0              ;RETURN
```

```
                        .SBTTL   *COMMAND ISSUED BY DRIVER SERVICE ROUTINE

                        ;;*****************************************************************
                        ;*
                        ;*      THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
                        ;*      AND CHECK IF A CONTROLLER ERROR OCCURRED.  IF A CONTROLLER
                        ;*      ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
                        ;*      CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
                        ;*      ADDRESS IN A.CONT.
                        ;*
                        ;*      REGISTER        USE
                        ;*      --------        ---
                        ;*
                        ;*      R2              ADDDRESS OF RK06 REGISTERS
                        ;*      R5              ADDRESS OF PARAMETER BLOCK
                        ;*
                        ;*CALL  JSR     R0,I.ISSU
                        ;*      <ADDRESS OF ERROR RETURN>
                        ;*      RETURN
                        ;*
                        ;*      ROUTINES USED:
                        ;*      --------------
                        ;*
                        ;*              I.CCLR
                        ;*              I.STOR
                        ;*
                        ;;*****************************************************************

036406 013746 003000            I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
036412 005037 003002                    CLR     T.CS2           ;CLEAR TEMPORARY CS2
036416 116537 000000 003002             MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
036424 013762 003002 000010             MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
036432 116537 000007 003001             MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
036440 142737 177753 003001             BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                                                ;  FORMAT AND DRIVE TYPE
036446 013762 003000 000000             MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
036454 105762 000000            1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
036460 100375                           BPL     1$
036462 004737 036630                    JSR     PC,I.STOR       ;GO STORE REGISTERS
036466 032737 100000 003000             BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
036474 001437                           BEQ     5$              ;NO, RETURN
036476 032737 001000 003002             BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
036504 001406                           BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
036506 052737 100000 003052             BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
036514 004737 037334                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
036520 000440                           BR      10$             ;RETURN

                                ;CHECK IF ANY CONTROLLER ERROR IS SET
036522 032737 024000 003000     2$:     BIT     #CTO!SPAR,T.CS1
036530 001027                           BNE     7$
036532 032737 176400 003002             BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
036540 001023                           BNE     7$
036542 032737 131761 003016             BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
036550 001017                           BNE     7$

036552 122716 000005                    CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
 7509  036556  001003                        BNE     3$              ;NO, DO NOT SET DRIVE HARD ERROR
 7510  036560  052765  000020  000014        BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
 7511  036566  004037  036324          3$:   JSR     R0,I.CCLR       ;GO ISSUE A CONTROLLER CLEAR
 7512  036572  036622                        IOS                     ;ERROR RETURN
 7513  036574  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
 7514  036602  005726                        TST     (SP)+           ;ADJUST STACK
 7515  036604  005720                        TST     (R0)+           ;ADJUST R0 FOR NORMAL RETURN
 7516  036606  000200                        RTS     R0              ;RETURN
 7517
 7518  036610  052737  001000  003052  7$:   BIS     #E.CERR,E.CONT  ;SET CONTROLLER ERROR DURING
 7519                                                                ;  DRIVER SERVICING
 7520  036616  004737  037334                JSR     PC,R.CONT       ;REPORT ERROR
 7521  036622  005726                  10$:  TST     (SP)+           ;ADJUST STACK
 7522  036624  011000                        MOV     (R0),R0         ;ADJUST R0 FOR ERROR RETURN
 7523  036626  000200                        RTS     R0              ;RETURN
```

```
                                .SBTTL   *STORE RK611 UNIBUS REGISTERS

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL
                                ;*      RK611 REGISTER IN TEMPORARY LOCATIONS.
                                ;*
                                ;*CALL  JSR     PC,I.STOR
                                ;*      RETURN
                                ;*
                                ;*      REGISTER        USE
                                ;*      --------        ---
                                ;*
                                ;*      R2              ADDRESS OF RK611 REGISTERS
                                ;*
                                ;;*****************************************************************

036630  016237  000000  003000  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
036636  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
036644  016237  000002  003004          MOV     RKWC(R2),T.WCR
036652  016237  000004  003006          MOV     RKBA(R2),T.BA
036660  016237  000006  003010          MOV     RKDA(R2),T.DA
036666  016237  000012  003020          MOV     RKDS(R2),T.DS
036674  016237  000014  003016          MOV     RKER(R2),T.ER
036702  016237  000016  003014          MOV     RKASOF(R2),T.ASOF
036710  016237  000020  003012          MOV     RKDCYL(R2),T.DC
036716  016237  000026  003022          MOV     RKMR1(R2),T.MR1
036724  016237  000034  003024          MOV     RKMR2(R2),T.MR2
036732  016237  000036  003026          MOV     RKMR3(R2),T.MR3
036740  016237  000030  003030          MOV     RKECPS(R2),T.POS
036746  016237  000032  003032          MOV     RKECPT(R2),T.PAT
036754  000207                          RTS     PC              ;RETURN
```

```
                                .SBTTL   *STORE CONTROLLER STATUS

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
                                ;*      THE FOLLOWING REGISTERS WILL BE STORED:
                                ;*
                                ;*              COMMAND AND STATUS REGISTER 2
                                ;*              WORD COUNT REGISTER
                                ;*              BUS ADDRESS REGISTER
                                ;*              DESIRED TRACK AND SECTOR
                                ;*              STATUS REGISTER
                                ;*              ERROR REGISTER
                                ;*              ATTENTION SUMMARY/OFFSET REGISTER
                                ;*              CYLINDER ADDRESS REGISTER
                                ;*
                                ;*CALL  JSR     PC,I.CSTS
                                ;*      RETURN
                                ;*
                                ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
                                ;*
                                ;*              REGISTER                CONTENTS
                                ;*              --------                --------
                                ;*
                                ;*              R2                      RK06 BASE ADDRESS
                                ;*              R5                      ADDRESS OF PARAMETER BLOCK
                                ;*
                                ;;*****************************************************************

036756  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
                                                                        ;OF LAST COMMAND ISSUED
036764  042737  000036  003000          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
036772  053765  003000  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
037000  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
037006  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
037014  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
037022  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
037030  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
037036  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
037044  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
                                                                        ;  OFFSET
037052  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
037060  000207                  RTS     PC                              ;RETURN
```

```
                        .SBTTL   *GATHER DRIVE STATUS

                        ;;*****************************************************************
                        ;*
                        ;*      THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
                        ;*      BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
                        ;*      HAS PREVIOUSLY BEEN SEIZED.  IT RUNS AT PRIORITY 7.
                        ;*
                        ;*CALL  JSR     R0,I.STAT
                        ;*      <ADDRESS OF ERROR RETURN>
                        ;*      RETURN
                        ;*
                        ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
                        ;*
                        ;*              REGISTER                CONTENTS
                        ;*              --------                --------
                        ;*
                        ;*              R2                      RK06 BASE ADDRESS
                        ;*              R5                      ADDRESS OF PARAMETER BLOCK
                        ;*
                        ;*      ROUTINES USED:
                        ;*              I.ISSU
                        ;*
                        ;;*****************************************************************

 037062  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 01
 037070  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 037076  004037  036406                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
 037102  037272                          3$                              ;ERROR RETURN
 037104  013765  003024  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
 037112  013765  003026  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
 037120  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 10
 037126  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 037134  004037  036406                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
 037140  037272                          3$                              ;ERROR RETURN
 037142  013765  003024  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
 037150  013765  003026  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
 037156  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
                                                                        ;  FOR STATUS BYTE 11
 037164  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 037172  004037  036406                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
 037176  037272                          3$                              ;ERROR RETURN
 037200  013765  003024  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
 037206  013765  003026  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
 037214  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 00
 037220  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 037226  004037  036406                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
 037232  037272                          3$                              ;ERROR RETURN
 037234  013765  003024  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
 037242  013765  003026  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
 037250  032737  001000  003026          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
 037256  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
 7657  037260  052737  002000  003052          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
 7658  037266  004737  037334                  JSR     PC,R.CONT               ;REPORT ERROR
 7659  037272  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
 7660  037274  000200                          RTS     R0                      ;RETURN
 7661
 7662  037276  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
 7663  037304  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
 7664  037306  000200                          RTS     R0                      ;RETURN
 7665
```

```
                                        .SBTTL  *COMMON DRIVER RETURNS

7666
7667
7668  037310  105037  003101  R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7669  037314  004777  143526          JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
7670  037320  000207                  RTS     PC              ;RETURN
7671
7672  037322  105037  003101  R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7673  037326  004777  143512          JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
7674  037332  000207                  RTS     PC              ;RETURN
7675
7676  037334  105037  003101  R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
7677  037340  105037  003100          CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
7678  037344  005037  003114          CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
7679  037350  004777  143474          JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
7680  037354  000207                  RTS     PC              ;RETURN
```

```
                                .SBTTL   #COMMAND INITATOR

                                ;;*****************************************************************
                                ;*
                                ;*      THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
                                ;*      BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
                                ;*      SPECIAL COMMAND ARE ALSO EXECUTED:
                                ;*
                                ;*              RELEASE
                                ;*              CONROLLER CLEAR
                                ;*              SUBSYSTEM CLEAR
                                ;*              READ ALL DRIVE STATUS
                                ;*              READ SPECIFIED HEADER
                                ;*
                                ;*      THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS
                                ;*
                                ;*CALL  JSR     PC,C.INIT
                                ;*      <ADDRESS OF PARAMETER BLOCK>
                                ;*      RETURN
                                ;*
                                ;*      FOR THE SEQUENTIAL OPERATIONS, THE DRIVER WILL LOAD THE
                                ;*      LOCATIONS, PBLKT AND INTMSK.
                                ;*
                                ;*      ROUTINES USED:
                                ;*              W.WTCH
                                ;*              I.CSTS
                                ;*              I.STAT
                                ;*              I.CCLR
                                ;*
                                ;;*****************************************************************

037356  010546                  C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
037360  010446                          MOV     R4,-(SP)        ;STORE R4 ON STACK
037362  010346                          MOV     R3,-(SP)        ;STORE R3 ON STACK
037364  010246                          MOV     R2,-(SP)        ;STORE R2 ON STACK
037366  010146                          MOV     R1,-(SP)        ;STORE R1 ON STACK
037370  010046                          MOV     R0,-(SP)        ;STORE R0 ON STACK
037372  013746  177776                  MOV     PS,-(SP)        ;STORE PSW ON STACK
037376  013737  003042  177776          MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
037404  017605  000016                  MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
037410  062766  000002  000016          ADD     #2,16(SP)       ;ADJUST RETURN
037416  016504  000000                  MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
037422  042704  177770                  BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
037426  010537  003112                  MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
037432  116437  003102  003101          MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
037440  116437  003102  003100          MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
037446  013737  003062  003114          MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

037454  013702  003036                  MOV     RKBAS,R2        ;LOAD R2 WITH RK06 ADDRESS BASE

                                ;       RESET ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
                                ;               DRIVE IN USE
                                ;               WRITE FOR WRITE CHECK
                                ;               NO CHECK
                                ;               DROP DRIVE FROM TEST SEQUENCE
                                ;               INHIBIT BUS ADDRESS INCREMENT
```

```
7737  037460  042765  075176  000014          BIC     #↑C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
7738
7739  037466  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
7740  037470  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
7741  037474  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
7742  037476  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
7743
7744  037502  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
7745  037504  020001                          CMP     R0,R1           ;CHECK IF FINISHED
7746  037506  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
7747  037510  105037  003074                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
7748  037514  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
7749  037520  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
7750  037524  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
7751  037532  001402                          BEQ     3$              ;NO, PROCESS
7752  037534  000137  040250                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
7753
7754  037540  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
7755                                                                  ;  START SPINDLE
7756                                                                  ;  RECALIBRATE
7757                                                                  ;  OFFSET
7758                                                                  ;  SEEK
7759                                                                  ;  UNLOAD
7760
7761  037546  101174                          BHI     25$             ;NO, DRIVE COMMAND
7762                                                                  ;  SELECT DRIVE
7763                                                                  ;  PACK ACKNOWLEDGE
7764                                                                  ;  CLEAR
7765
7766  037550  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
7767  037556  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
7768                                                                  ;  READ DATA
7769                                                                  ;  WRITE DATA
7770                                                                  ;  READ HEADER
7771                                                                  ;  WRITE HEADER
7772                                                                  ;  WRITE CHECK
7773  037560  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
7774  037566  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
7775  037574  005037  003054                  CLR     @.WAIT          ;CLEAR WAIT FOR COMMAND
7776  037600  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
7777  037606  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
7778  037610  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
7779  037616  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
7780  037624  000431                          BR      8$              ;GO ISSUE COMMAND
7781
7782  037626  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
7783  037634  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
7784  037636  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
7785  037644  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
7786  037652  000416                          BR      8$              ;GO ISSUE COMMAND
7787
7788  037654  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
7789  037662  001003                          BNE     7$              ;NO, CHECK IF RECAL
7790  037664  013737  003066  003114          MOV     W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
7791  037672  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
7792  037700  001003                          BNE     8$              ;NO, CONTINUE
```

```
7793  037702  013737  003064  003114          MOV     W.8SEC,W.DRV    ;LOAD RECAL TIME FOR 8 SECONDS
7794  037710  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7795  037716  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
7796                                                  ;  AND DRIVE TYPE
7797  037724  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
7798  037732  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
7799  037740  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
7800  037746  001533                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
7801  037750  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
7802  037756  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
7803  037764  004737  033726          10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
7804  037770  016237  000000  003000          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7805  037776  032737  000200  003000          BIT     #RDY,T.CS1      ;WAIT FOR READY
7806  040004  001767                          BEQ     10$
7807  040006  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK FOR ERROR
7808  040014  001011                          BNE     15$             ;YES, GIVE NORMAL RETURN
7809  040016  004737  033726          11$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
7810  040022  016237  000016  003014          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7811  040030  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
7812  040036  001767                          BEQ     11$             ;WAIT FOR DRIVE INTERRUPT
7813  040040  105037  003100          15$:    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
7814  040044  005037  003114                  CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
7815  040050  004737  037322                  JSR     PC,R.NORM       ;INDICATE COMMAND IS FINISHED
7816  040054  000137  041230                  JMP     C.RTRN          ;RESTORE REGISTERS
7817
7818  040060  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
7819  040066  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
7820  040074  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
7821  040102  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
7822  040110  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
7823  040116  001010                          BNE     25$             ;NO, GO ISSUE THE COMMAND
7824  040120  032765  000200  000014          BIT     #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
7825  040126  001404                          BEQ     25$             ;NO, GO ISSUE THE COMMAND
7826  040130  012765  000123  000016          MOV     #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
7827  040136  000406                          BR      26$             ;GO ISSUE COMMAND
7828
7829  040140  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
7830  040146  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
7831  040154  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7832  040162  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
7833                                                  ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
7834                                                  ;  BITS 16-17
7835  040170  010537  003054                  MOV     R5,O.WAIT       ;LOAD WAITING FOR COMMAND
7836  040174  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
7837  040202  001403                          BEQ     27$             ;NO, LOAD CS2
7838  040204  052765  000020  000020          BIS     #BAI,P.CS2(R5)  ;SET INHIBIT BUS ADDRESS INCREMENT
7839  040212  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
7840  040220  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
7841  040226  001403                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
7842  040230  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
7843  040236  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
7844  040244  000137  041230                  JMP     C.RTRN          ;RESTORE REGISTERS
7845
7846                                  .SBTTL  *SPECIAL COMMAND PROCESSING
7847
7848  040250  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
7849  040256  001132                        BNE     10$             ;NO, PROCESS OTHER COMMANDS
7850  040260  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
7851  040266  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7852  040274  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
7853                                                ;  AND DRIVE TYPE
7854  040302  112765  000001  000016        MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
7855  040310  016562  000016  000000        MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
7856  040316  004737  033726          2$:   JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
7857  040322  016265  000000  000016        MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
7858  040330  032765  000200  000016        BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
7859  040336  001767                        BEQ     2$
7860  040340  004737  037000                JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
7861  040344  016265  000034  000040        MOV     RKMR2(R2),P.AOO(R5) ;STORE STATUS BYTE OO MESSAGE A
7862  040352  016265  000036  000042        MOV     RKMR3(R2),P.BOO(R5) ;STORE STATUS BYTE OO MESSAGE B
7863  040360  032765  100000  000016        BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
7864  040366  001436                        BEQ     6$              ;NO, GATHER DRIVE STATUS
7865  040370  105037  003100                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
7866  040374  005037  003114                CLR     W.DRV           ;CLEAR WATCH DOG COUNT
7867  040400  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
7868  040406  001043                        BNE     8$              ;YES, INDICATE NORMAL RETURN
7869  040410  032765  001000  000020        BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
7870  040416  001043                        BNE     9$              ;YES, INDICATE CONTROLLER ERROR
7871  040420  004037  036324                JSR     R0,I.CCLR       ;CLEAR ERROR
7872  040424  041230                        C.RTRN                  ;ERROR RETURN
7873  040426  032765  010400  000020        BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
7874  040434  001007                        BNE     5$              ;REPORT ERROR
7875  040436  032765  000001  000036        BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
7876  040444  001003                        BNE     5$              ;YES, REPORT ERROR
7877  040446  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
7878  040454  004737  037310          5$:   JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
7879  040460  000137  041230                JMP     C.RTRN          ;RESTORE REGISTERS
7880
7881  040464  004037  037062          6$:   JSR     R0,I.STAT       ;GATHER DRIVE STATUS
7882  040470  041230                        C.RTRN                  ;ERROR RETURN
7883  040472  105037  003100                CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
7884  040476  005037  003114                CLR     W.DRV           ;RESET WATCH-DOG TIME
7885  040502  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
7886  040510  001402                        BEQ     8$              ;NO, REPORT ERROR
7887  040512  005062  000000                CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
7888  040516  004737  037322          8$:   JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
7889  040522  000137  041230                JMP     C.RTRN          ;RESTORE REGISTERS
7890
7891  040526  052737  100000  003052  9$:   BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
7892  040534  004737  037334                JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
7893  040540  000137  041230                JMP     C.RTRN
7894
7895  040544  122765  000140  000001  10$:   CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
7896  040552  001040                        BNE     13$             ;NO, CHECK IF READ ALL HEADERS
7897  040554  010537  003054                MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
7898                                                ;  WAIT FOR COMMAND
7899  040560  052765  000010  000020        BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
7900  040566  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
7901  040574  112737  000001  003074        MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
7902  040602  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
7903  040610  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
7904                                                ;  AND DRIVE TYPE
```

```
7905  040616  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
7906  040624  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
7907  040632  001403                          BEQ     11$             ;NO, DO NOT RESET INTERRUPT ENABLE
7908  040634  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;RESET INTERRUPT ENABLE
7909  040642  016562  000016  000000  11$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
7910  040650  000137  041230                  JMP     C.RTRN          ;RESTORE REGISTERS
7911
7912  040654  122765  000164  000001  13$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
7913  040662  001053                          BNE     30$             ;NO, CHECK IF CONTROLLER CLEAR
7914  040664  010537  003054                  MOV     R5,Q.WAIT       ;SET WAITING FOR COMMAND COMMPLETION
7915  040670  016537  000010  003070          MOV     P.BAL0(R5),HDR.AD ;LOAD HEADER ADDRESS
7916  040676  132765  000020  000007          BITB    #8.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
7917  040704  001404                          BEQ     14$             ;YES, LOAD 22 IN HEADER COUNT
7918  040706  012737  000024  003072          MOV     #20.,HDR.CT     ;LOAD 20 IN SECTOR COUNT
7919  040714  000403                          BR      22$             ;GO ISSUE READ HEADER COMMAND
7920
7921  040716  012737  000026  003072  14$:    MOV     #22.,HDR.CT     ;LOAD 22 IN SECTOR COUNT
7922  040724  016562  000002  000020  22$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
7923  040732  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
7924  040740  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
7925  040746  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
7926  040754  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
7927                                                  ;  AND FORMAT
7928  040762  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
7929  040770  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
7930  040776  001027                          BNE     34$             ;YES, INDICATE ILLEGAL DRIVER COMMAND
7931  041000  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
7932  041006  000137  041230                  JMP     C.RTRN          ;RESTORE REGISTERS
7933
7934  041012  122765  000176  000001  30$:    CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
7935  041020  001012                          BNE     32$             ;NO, CHECK IF SUBSYSTEM CLEAR
7936  041022  004037  036324                  JSR     R0,I.CCLR       ;CLEAR CONTROLLER
7937  041026  041230                          C.RTRN                  ;ERROR RETURN
7938  041030  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
7939  041036  001472                          BEQ     40$             ;NO, INDICATE NORMAL RETURN
7940  041040  005062  000000                  CLR     RKCS1(R2)       ;RESET INTERRUPT ENABLE
7941  041044  000467                          BR      40$             ;INDICATE NORMAL RETURN
7942
7943  041046  122765  000177  000001  32$:    CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
7944  041054  001406                          BEQ     36$             ;YES, CLEAR SUBSYSTEM
7945  041056  052737  000100  003052  34$:    BIS     #E.ILLD,E.CONT  ;SET ILLEGAL DRIVER COMMAND
7946  041064  004737  037334                  JSR     PC,R.CONT       ;REPORT ERROR
7947  041070  000457                          BR      C.RTRN          ;RESTORE REGISTERS
7948
7949  041072  012762  000040  000010  36$:    MOV     #SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
7950  041100  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
7951  041106  032765  100000  000016          BIT     #CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
7952  041114  001406                          BEQ     37$             ;NO, FINISH COMMAND
7953  041116  052737  000001  003052          BIS     #BIT0,E.CONT    ;SET CLEAR SUBSYSTEM DID NOT CLEAR
7954                                                  ;  CONTROLLER ERROR
7955  041124  004737  037334                  JSR     PC,R.CONT       ;REPORT ERROR
7956  041130  000437                          BR      C.RTRN          ;RESTORE REGISTERS
7957
7958  041132  013746  003060          37$:    MOV     W.MILI,-(SP)    ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
7959                                                  ;  TO DISAPPEAR
7960  041136  016265  000000  000016  38$:    MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
 7961  041144  032765  040000  000016          BIT     #DI,P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
 7962  041152  001411                          BEQ     39$             ;YES, FINISH COMMAND
 7963  041154  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
 7964  041156  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
 7965  041160  005726                          TST     (SP)+           ;ADJUST STACK
 7966  041162  052737  000040  003052          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
 7967                                                                  ;  DRIVE ATTENTIONS
 7968  041170  004737  037334                  JSR     PC,R.CONT       ;REPORT ERROR
 7969  041174  000415                          BR      C.RTRN          ;RESTORE REGISTER
 7970
 7971  041176  005726                  39$:    TST     (SP)+           ;ADJUST STACK
 7972  041200  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 7973  041206  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
 7974  041210  112737  177777  003074          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
 7975  041216  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
 7976  041224  004737  037322                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
 7977
 7978  041230  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
 7979  041234  012600                          MOV     (SP)+,R0        ;RESTORE R0
 7980  041236  012601                          MOV     (SP)+,R1        ;RESTORE R1
 7981  041240  012602                          MOV     (SP)+,R2        ;RESTORE R2
 7982  041242  012603                          MOV     (SP)+,R3        ;RESTORE R3
 7983  041244  012604                          MOV     (SP)+,R4        ;RESTORE R4
 7984  041246  012605                          MOV     (SP)+,R5        ;RESTORE R5
 7985  041250  000207                          RTS     PC              ;RETURN
 7986                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
 7987
 7988                                  ;;**********************************************************************
 7989                                  ;*
 7990                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
 7991                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
 7992                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
 7993                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
 7994                                  ;*
 7995                                  ;*CALL
 7996                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
 7997                                  ;*      JSR     PC,OCTBIN
 7998                                  ;*      <ADDRESS OF ERROR RETURN>
 7999                                  ;*      RETURN
 8000                                  ;*
 8001                                  ;;**********************************************************************
 8002
 8003  041252  010046                  OCTBIN: MOV     R0,-(SP)        ;SAVE R0
 8004  041254  010146                          MOV     R1,-(SP)        ;SAVE R1
 8005  041256  010246                          MOV     R2,-(SP)        ;SAVE R2
 8006  041260  016600  000010                  MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
 8007  041264  005001                          CLR     R1              ;CLEAR DATA WORDS
 8008  041266  005002                          CLR     R2
 8009  041270  112046                  2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
 8010  041272  001423                          BEQ     3$              ;IF ZERO GET OUT
 8011  041274  121627  000054                  CMPB    (SP),#',        ;CHECK IF COMMA
 8012  041300  001420                          BEQ     3$              ;IF COMMA GET OUT
 8013  041302  122716  000060                  CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
 8014  041306  003030                          BGT     4$              ;  AN OCTAL DIGIT
 8015  041310  122716  000067                  CMPB    #'7,(SP)
 8016  041314  002425                          BLT     4$
```

```
8017  041316  006301                        ASL     R1              ; *2
8018  041320  006102                        ROL     R2
8019  041322  006301                        ASL     R1              ; *4
8020  041324  006102                        ROL     R2
8021  041326  006301                        ASL     R1              ; *8
8022  041330  006102                        ROL     R2
8023  041332  042716  177770                BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
8024  041336  062601                        ADD     (SP)+,R1        ;ADD THIS DIGIT
8025  041340  000753                        BR      2$              ;LOOP
8026  041342  005726                3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
8027  041344  010166  000010                MOV     R1,10(SP)       ;SAVE RESULT
8028  041350  010237  041404                MOV     R2,$HIOCT
8029  041354  012602                        MOV     (SP)+,R2        ;RESTORE R2
8030  041356  012601                        MOV     (SP)+,R1        ;RESTORE R1
8031  041360  012600                        MOV     (SP)+,R0        ;RESTORE R0
8032  041362  062716  000002                ADD     #2,(SP)         ;ADJUST RETURN
8033  041366  000207                        RTS     PC              ;RETURN
8034
8035  041370  005726                4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
8036  041372  012602                        MOV     (SP)+,R2        ;RESTORE R2
8037  041374  012601                        MOV     (SP)+,R1        ;RESTORE R1
8038  041376  012600                        MOV     (SP)+,R0        ;RESTORE R0
8039  041400  013616                        MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
8040  041402  000207                        RTS     PC              ;GO PROCESS ERROR
8041  041404  000000                $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
8042                                 .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
8043
8044                                 ;;*****************************************************************
8045                                 ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
8046                                 ;*WITH A RANGE OF 0 TO 2(+33)-1.
8047                                 ;*CALL:
8048                                 ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
8049                                 ;*      RETURN                  ;;RETURN HERE THE RANDOM
8050                                 ;*                              ;;NUMBER WILL BE IN
8051                                 ;*                              ;;$HINUM,$LONUM
8052
8053  041406                         $RAND:
8054  041406  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8055  041410  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8056  041412  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8057  041414  013700  041506                MOV     $LONUM,R0       ;;SET R0 WITH LOW
8058  041420  013701  041504                MOV     $HINUM,R1       ;;SET R1 WITH HIGH
8059  041424  012702  177771                MOV     #-7,R2          ;;SET SHIFT COUNT
8060  041430  006300                1$:     ASL     R0              ;;SHIFT R0 LEFT AND
8061  041432  006101                        ROL     R1              ;;ROTATE CARRY INTO R1 AND
8062  041434  005202                        INC     R2              ;;CHECK FOR DONE
8063  041436  001374                        BNE     1$              ;;CONTINUE SHIFT LOOP
8064  041440  063700  041506                ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
8065  041444  005501                        ADC     R1              ;;PROPOGATE CARRY
8066  041446  063701  041504                ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
8067  041452  062700  001057                ADD     #1057,R0        ;;ADD LOW CONSTANT
8068  041456  005501                        ADC     R1              ;;PROPOGATE CARRY
8069  041460  062701  047401                ADD     #47401,R1       ;;ADD HIGH CONSTANT
8070  041464  010037  041506                MOV     R0,$LONUM       ;;SAVE R0
8071  041470  010137  041504                MOV     R1,$HINUM       ;;SAVE R1
8072  041474  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
8073  041476  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
8074  041500  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
8075  041502  000207                        RTS     PC              ;;RETURN
8076  041504  176543                $HINUM: .WORD   176543
8077  041506  123456                $LONUM: .WORD   123456
8078                                 .SBTTL  TYPE ROUTINE
8079
8080                                 ;;*****************************************************************
8081                                 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8082                                 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8083                                 ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
8084                                 ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
8085                                 ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
8086                                 ;*
8087                                 ;*CALL:
8088                                 ;*1) USING A TRAP INSTRUCTION
8089                                 ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
8090                                 ;*OR
8091                                 ;*      TYPE
8092                                 ;*      MESADR
8093                                 ;*
8094
8095  041510  105737  001157         $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
8096  041514  100002                         BPL     1$              ;;BR IF YES
8097  041516  000000                         HALT                    ;;HALT HERE IF NO TERMINAL
8098  041520  000430                         BR      3$              ;;LEAVE
8099  041522  010046                 1$:     MOV     R0,-(SP)        ;;SAVE R0
8100  041524  017600  000002                 MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
8101  041530  122737  000001  001350         CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
8102  041536  001011                         BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
8103  041540  132737  000100  001351         BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
8104  041546  001405                         BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
8105  041550  010037  041560                 MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
8106  041554  004737  044120                 JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
8107  041560  000000                 61$:    .WORD   0               ;;MESSAGE ADDRESS
8108  041562  132737  000040  001351 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
8109  041570  001003                         BNE     60$             ;;YES,SKIP TYPE OUT
8110  041572  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
8111  041574  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
8112  041576  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
8113  041600  012600                 60$:    MOV     (SP)+,R0        ;;RESTORE R0
8114  041602  062716  000002         3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
8115  041606  000002                         RTI                     ;;RETURN
8116  041610  122716  000011         4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
8117  041614  001430                         BEQ     8$
8118  041616  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
8119  041622  001006                         BNE     5$
8120  041624  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
8121  041626  104401                         TYPE                    ;;TYPE A CR AND LF
8122  041630  001325                         $CRLF
8123  041632  105037  041766                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
8124  041636  000755                         BR      2$              ;;GET NEXT CHARACTER
8125  041640  004737  041722         5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
8126  041644  123726  001156         6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
8127  041650  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
8128  041652  013746  001154                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
8129
8130  041656  105366  000001          7$:     DECB    1(SP)           ;;AND THE NULL CHAR,
8131  041662  002770                          BLT     6$              ;;DOES A NULL NEED TO BE TYPED?
8132  041664  004737  041722                  JSR     PC,$TYPEC       ;;BR IF NO--GO POP THE NULL OFF OF STACK
8133  041670  105337  041766                  DECB    $CHARCNT        ;;GO TYPE A NULL
8134  041674  000770                          BR      7$              ;;DO NOT COUNT AS A COUNT
8135                                                                  ;;LOOP
8136                                  ;HORIZONTAL TAB PROCESSOR
8137
8138  041676  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
8139  041702  004737  041722          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
8140  041706  132737  000007  041766          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
8141  041714  001372                          BNE     9$              ;;TAB STOP
8142  041716  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
8143  041720  000724                          BR      2$              ;;GET NEXT CHARACTER
8144  041722  105777  137222          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
8145  041726  100375                          BPL     $TYPEC
8146  041730  116677  000002  137214          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8147  041736  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
8148  041744  001003                          BNE     1$              ;;BRANCH IF NO
8149  041746  105037  041766                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
8150  041752  000406                          BR      $TYPEX          ;;EXIT
8151  041754  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
8152  041762  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
8153  041764  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
8154  041766  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
8155  041770  000207          $TYPEX: RTS     PC
8156
8157
8158
8159
8160                                  ;;*****************************************************************
8161                                  .SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
8162                                  ;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
8163                                  ;*      USES ALL REGISTERS (R0-R5)
8164                                  ;*
8165                                  ;*      ENTER WITH   JSR   PC,M.DPIM
8166                                  ;*                   MULTIPLIER IN R2-R3
8167                                  ;*                   MULTIPLICAND IN R4-R5
8168                                  ;*                   PRODUCT RETURNED IN R0-R1-R2-R3
8169                                  ;*
8170                                  ;;*****************************************************************
8171  041772  005000          M.DPIM: CLR     R0              ;CLEAR HI ORDER WORDS
8172  041774  005001                  CLR     R1
8173  041776  012746  000041          MOV     #41,-(SP)       ;MOVE 33 (DEC) TO COUNTER
8174  042002  006000          M.DP01: ROR     R0
8175  042004  006001                  ROR     R1
8176  042006  006002                  ROR     R2              ;SHIFT TO ADD
8177  042010  006003                  ROR     R3
8178  042012  103003                  BCC     M.DP02          ;NO CARRY NO ADD
8179  042014  060501                  ADD     R5,R1
8180  042016  005500                  ADC     R0              ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
8181  042020  060400                  ADD     R4,R0           ;  PRODUCT
8182  042022  005316          M.DP02: DEC     @SP             ;DECREMENT COUNTER
8183  042024  001366                  BNE     M.DP01
8184  042026  005726                  TST     (SP)+           ;REMOVE THE COUNTER
```

```
                                        RTS     PC


                                ;;*****************************************************************
                                .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
                                ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
                                ;*              USES ALL REGISTERS (R0-R5)
                                ;*
                                ;*              ENTER WITH  JSR   PC,M.DPID
                                ;*                      DIVIDEND IN R0-R1-R2-R3
                                ;*                      DIVISOR IN R4-R5
                                ;*                      REMAINDER RETURNED IN R0-R1
                                ;*                      QUOTIENT RETURNED IN R2-R3
                                ;*
                                ;*****************************************************************
042032  012746  000040          M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
042036  010446                          MOV     R4,-(SP)        ;HI ORDER
042040  010546                          MOV     R5,-(SP)        ;LO ORDER DIVISOR TO THE STACK
042042  005466  000002                  NEG     2(SP)           ;FORM NEGATIVE
042046  005416                          NEG     @SP             ;   VERSION OF THE DIVISOR
042050  005666  000002                  SBC     2(SP)
042054  061601                          ADD     @SP,R1
042056  005500                          ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
042060  066600  000002                  ADD     2(SP),R0
042064  103445                          BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
042066  005046                          CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
042070  006103                  M.DP40: ROL     R3
042072  006102                          ROL     R2
042074  006101                          ROL     R1
042076  006100                          ROL     R0
042100  005716                          TST     @SP             ;TEST "CARRY INDICATOR"
042102  001410                          BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
042104  005016                          CLR     @SP             ;CLEAR UP FOR NEXT TIME
042106  066601  000002                  ADD     2(SP),R1
042112  005500                          ADC     R0              ;ADD -(DIVISOR)
042114  005516                          ADC     @SP     ; I     ;SET "CARRY"
042116  066600  000004                  ADD     4(SP),R0 ;<
042122  000404                          BR      M.DP42
042124  060501                  M.DP41: ADD     R5,R1
042126  005500                          ADC     R0              ;ADD +(DIVISOR)
042130  005516                          ADC     @SP     ; I     ;SET "CARRY"
042132  060400                          ADD     R4,R0   ;<
042134  005516                  M.DP42: ADC     @SP             ;SET "CARRY"
042136  005716                          TST     @SP             ;TEST THE UPDATE INDICATOR
042140  001401                          BEQ     .+4     ;>      ;IF ZERO FORGET IT
042142  005203                          INC     R3      ; I     ;NO CARRY POSSIBLE HERE
042144  005366  000006                  DEC     6(SP)   ;<      ;DECREMENT COUNTER
042150  003347                          BGT     M.DP40          ;BR IF MORE TO DO
042152  006003                          ROR     R3
042154  103404                          BCS     M.DP44
042156  060501                          ADD     R5,R1
042160  005500                          ADC     R0
042162  060400                          ADD     R4,R0
042164  000241                          CLC
042166  006103                  M.DP44: ROL     R3
042170  062706  000010                  ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
```

```
8241  042174  000242                      CLV
8242  042176  000207                      RTS     PC
8243  042200  062706  000006    M.DP50:   ADD     #6,SP
8244  042204  000262                      SEV
8245  042206  000207                      RTS     PC
8246
8247
8248                                      .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
8249
8250                                      ;;**********************************************************************
8251                                      ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
8252                                      ;*UNSIGNED OCTAL ASCIZ NUMBER.
8253                                      ;*CALL
8254                                      ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
8255                                      ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
8256                                      ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
8257
8258
8259
8260  042210  104407            $DB2O:    SAVREG                          ;;SAVE ALL REGISTERS
8261  042212  016601  000002              MOV     2(SP),R1                ;;PICKUP THE POINTER TO LOW WORD
8262  042216  012705  042327              MOV     #$OCTVL+13.,R5          ;;POINTER TO DATA TABLE
8263  042222  012704  000014              MOV     #12.,R4                 ;;DO ELEVEN CHARACTERS
8264  042226  012703  177770              MOV     #^C7,R3                 ;;MASK
8265  042232  012100                      MOV     (R1)+,R0                ;;LOWER WORD
8266  042234  012101                      MOV     (R1)+,R1                ;;HIGH WORD
8267  042236  005002                      CLR     R2                      ;;TERMINATOR
8268  042240  110245            1$:       MOVB    R2,-(R5)                ;;PUT CHARACTER IN DATA TABLE
8269  042242  010002                      MOV     R0,R2                   ;;GET THIS DIGIT
8270  042244  005304                      DEC     R4                      ;;COUNT THIS CHARACTER
8271  042246  003007                      BGT     3$                      ;;BR IF NOT THE LAST DIGIT
8272  042250  001405                      BEQ     2$                      ;;BR IF IT IS THE LAST DIGIT
8273  042252  005205                      INC     R5                      ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
8274  042254  010566  000002              MOV     R5,2(SP)                ;;ASCIZ CHAR. & PUT IT ON THE STACK
8275  042260  104410                      RESREG                          ;;RESTORE ALL REGISTERS
8276  042262  000207                      RTS     PC                      ;;RETURN TO USER
8277  042264  006203            2$:       ASR     R3                      ;;POSITION THE MASK FOR THE LAST DIGIT
8278  042266  006001            3$:       ROR     R1                      ;;POSITION THE BINARY NUMBER FOR
8279  042270  006000                      ROR     R0                      ;;      THE NEXT OCTAL DIGIT
8280  042272  006001                      ROR     R1
8281  042274  006000                      ROR     R0
8282  042276  006001                      ROR     R1
8283  042300  006000                      ROR     R0
8284  042302  040302                      BIC     R3,R2                   ;;MASK OUT ALL JUNK
8285  042304  062702  000060              ADD     #'0,R2                  ;;MAKE THIS CHAR. ASCII
8286  042310  000753                      BR      1$                      ;;GO PUT IT IN THE DATA TABLE
8287  042312  000016            $OCTVL:   .BLKB   14.                     ;;RESERVE DATA TABLE
8288                                      .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
8289
8290                                      ;;**********************************************************************
8291                                      ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
8292                                      ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
8293                                      ;*POSITIVE.
8294                                      ;*CALL
8295                                      ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
8296                                      ;*      JSR     PC,@#$DB2D
```

```
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK

                                        $DB2D:  SAVREG                  ;;SAVE REGISTERS
042330  104407
042332  016602  000002                        MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
042336  012700  042510                        MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
042342  010066  000002                        MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
042346  012201                              MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
042350  012202                              MOV     (R2)+,R2
042352  012737  000012  042426              MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
042360  012704  042440                      MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
042364  012705  042442                      MOV     #$TNPWR+2,R5
042370  005003                        1$:   CLR     R3              ;;CLEAR PARTIAL
042372  161401                        2$:   SUB     (R4),R1         ;;SUBTRACT TEN POWER
042374  005602                              SBC     R2
042376  161502                              SUB     (R5),R2
042400  002402                              BLT     3$              ;;BR IF TEN POWER TO LARGE
042402  005203                              INC     R3              ;;ADD 1 TO PARTIAL
042404  000772                              BR      2$              ;;LOOP
042406  062401                        3$:   ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
042410  005502                              ADC     R2
042412  062402                              ADD     (R4)+,R2
042414  022525                              CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
042416  052703  000060                      BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
042422  110320                              MOVB    R3,(R0)+        ;;SAVE IT
042424  005327                              DEC     (PC)+           ;;DONE?
042426  000000                        4$:   .WORD   0
042430  001357                              BNE     1$              ;;BR IF NO
042432  105020                              CLRB    (R0)+           ;;TERMINATOR
042434  104410                              RESREG                  ;;RESTORE REGISTERS
042436  000207                              RTS     PC              ;;RETURN
042440  145000                  STNPWR: 145000                        ;;1.0E09
042442  035632                          35632
042444  160400                          160400                      ;;1.0E08
042446  002765                          2765
042450  113200                          113200                      ;;1.0E07
042452  000230                          230
042454  041100                          041100                      ;;1.0E06
042456  000017                          17
042460  103240                          103240                      ;;1.0E05
042462  000001                          1
042464  023420                          23420                       ;;1.0E04
042466  000000                          0
042470  001750                          1750                        ;;1.0E03
042472  000000                          0
042474  000144                          144                         ;;1.0E02
042476  000000                          0
042500  000012                          12                          ;;1.0E01
042502  000000                          0
042504  000001                          1                           ;;1.0E00
042506  000000                          0
042510  000014                  $DECVL: .BLKB   12.                  ;;RESERVE STORAGE FOR ASCIZ STRING
                                .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

```
 8353                                   ;#THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
 8354                                   ;#LEADING NUMBERS.
 8355                                   ;#CALL
 8356                                   ;#       MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
 8357                                   ;#       JSR     PC,@#$SUPRS
 8358
 8359
 8360  042524  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
 8361  042526  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
 8362  042532  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
 8363  042534  001403                          BEQ     2$              ;;BR IF YES
 8364  042536  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
 8365  042542  001773                          BEQ     1$              ;;BR IF YES
 8366  042544  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
 8367  042546  010037  042554                  MOV     R0,3$           ;;SAVE FOR TYPING
 8368  042552  104401                          TYPE                    ;;GO TYPE
 8369  042554  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
 8370  042556  012600                          MOV     (SP)+,R0        ;;RESTORE R0
 8371  042560  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
 8372  042562  000207                          RTS     PC              ;;RETURN
 8373                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 8374
 8375                                   ;;*****************************************************************
 8376                                   ;#THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 8377                                   ;#OCTAL (ASCII) NUMBER AND TYPE IT.
 8378                                   ;#$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 8379                                   ;#CALL:
 8380                                   ;#       MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 8381                                   ;#       TYPOS                   ;;CALL FOR TYPEOUT
 8382                                   ;#       .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 8383                                   ;#       .BYTE   M               ;;M=1 OR 0
 8384                                   ;#                                       ;;1=TYPE LEADING ZEROS
 8385                                   ;#                                       ;;0=SUPPRESS LEADING ZEROS
 8386                                   ;#
 8387                                   ;#$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 8388                                   ;#$TYPOS OR $TYPOC
 8389                                   ;#CALL:
 8390                                   ;#       MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 8391                                   ;#       TYPON                   ;;CALL FOR TYPEOUT
 8392                                   ;#
 8393                                   ;#$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 8394                                   ;#CALL:
 8395                                   ;#       MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 8396                                   ;#       TYPOC                   ;;CALL FOR TYPEOUT
 8397
 8398  042564  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 8399  042570  116637  000001  043007          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 8400  042576  112637  043011                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 8401  042602  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 8402  042606  000406                          BR      $TYPON
 8403  042610  112737  000001  043007  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 8404  042616  112737  000006  043011          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 8405  042624  112737  000005  043006  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 8406  042632  010346                          MOV     R3,-(SP)        ;;SAVE R3
 8407  042634  010446                          MOV     R4,-(SP)        ;;SAVE R4
 8408  042636  010546                          MOV     R5,-(SP)        ;;SAVE R5
```

```
8409  042640  113704  043011              MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
8410  042644  005404                      NEG     R4
8411  042646  062704  000006              ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
8412  042652  110437  043010              MOVB    R4,$OMODE       ;;SAVE IT FOR USE
8413  042656  113704  043007              MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
8414  042662  016605  000012              MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
8415  042666  005003                      CLR     R3              ;;CLEAR THE OUTPUT WORD
8416  042670  006105              1$:     ROL     R5              ;;ROTATE MSB INTO "C"
8417  042672  000404                      BR      3$              ;;GO DO MSB
8418  042674  006105              2$:     ROL     R5              ;;FORM THIS DIGIT
8419  042676  006105                      ROL     R5
8420  042700  006105                      ROL     R5
8421  042702  010503                      MOV     R5,R3
8422  042704  006103              3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
8423  042706  105337  043010              DECB    $OMODE          ;;TYPE THIS DIGIT?
8424  042712  100016                      BPL     7$              ;;BR IF NO
8425  042714  042703  177770              BIC     #177770,R3      ;;GET RID OF JUNK
8426  042720  001002                      BNE     4$              ;;TEST FOR 0
8427  042722  005704                      TST     R4              ;;SUPPRESS THIS 0?
8428  042724  001403                      BEQ     5$              ;;BR IF YES
8429  042726  005204              4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
8430  042730  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
8431  042734  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
8432  042740  110337  043004              MOVB    R3,8$           ;;SAVE FOR TYPING
8433  042744  104401  043004              TYPE    ,8$             ;;GO TYPE THIS DIGIT
8434  042750  105337  043006      7$:     DECB    $OCNT           ;;COUNT BY 1
8435  042754  003347                      BGT     2$              ;;BR IF MORE TO DO
8436  042756  002402                      BLT     6$              ;;BR IF DONE
8437  042760  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
8438  042762  000744                      BR      2$              ;;GO DO THE LAST DIGIT
8439  042764  012605              6$:     MOV     (SP)+,R5        ;;RESTORE R5
8440  042766  012604                      MOV     (SP)+,R4        ;;RESTORE R4
8441  042770  012603                      MOV     (SP)+,R3        ;;RESTORE R3
8442  042772  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
8443  043000  012616                      MOV     (SP)+,(SP)
8444  043002  000002                      RTI                     ;;RETURN
8445  043004     000              8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
8446  043005     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
8447  043006     000              $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
8448  043007     000              $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
8449  043010  000000              $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
8450                               .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
8451
8452                               ;;**************************************************************
8453                               ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
8454                               ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
8455                               ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
8456                               ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
8457                               ;*REPLACED WITH SPACES.
8458                               ;*CALL:
8459                               ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
8460                               ;*      TYPDS                   ;;GO TO THE ROUTINE
8461
8462  043012                       $TYPDS:
8463  043012  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8464  043014  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
8465  043016  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8466  043020  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8467  043022  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
8468  043024  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
8469  043030  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
8470  043034  100004                        BPL     1$              ;;BR IF INPUT IS POS.
8471  043036  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
8472  043040  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
8473  043046  005000                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
8474  043050  012703  043226                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
8475  043054  112723  000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
8476  043060  005002                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
8477  043062  016001  043216                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
8478  043066  160105                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
8479  043070  002402                        BLT     4$              ;;BR IF DONE
8480  043072  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
8481  043074  000774                        BR      3$
8482  043076  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
8483  043100  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
8484  043102  001002                        BNE     5$              ;;FALL THROUGH IF 0
8485  043104  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
8486  043106  100407                        BMI     7$              ;;BR IF YES
8487  043110  106316                5$:     ASLB    (SP)            ;;MSD?
8488  043112  103003                        BCC     6$              ;;BR IF NO
8489  043114  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
8490  043122  052702  000060        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
8491  043126  052702  000040        7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
8492  043132  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
8493  043134  005720                        TST     (R0)+           ;;JUST INCREMENTING
8494  043136  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
8495  043142  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
8496  043144  003002                        BGT     8$              ;;GO TO EXIT
8497  043146  010502                        MOV     R5,R2           ;;GET THE LSD
8498  043150  000764                        BR      6$              ;;GO CHANGE TO ASCII
8499  043152  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
8500  043154  100003                        BPL     9$              ;;BR IF NO
8501  043156  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
8502  043164  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
8503  043166  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
8504  043170  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
8505  043172  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
8506  043174  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
8507  043176  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
8508  043200  104401  043226                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
8509  043204  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
8510  043212  012616                        MOV     (SP)+,(SP)
8511  043214  000002                        RTI                     ;;RETURN TO USER
8512  043216  023420                $DTBL:  10000.
8513  043220  001750                        1000.
8514  043222  000144                        100.
8515  043224  000012                        10.
8516  043226  000004                $DBLK:  .BLKW   4
8517                                 .SBTTL  ERROR HANDLER ROUTINE
8518
8519                                 ;;*****************************************************************
8520                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

```
8521                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
8522                                    ;*AND GO TO TYPERR ON ERROR
8523                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
8524                                    ;*SW15=1        HALT ON ERROR
8525                                    ;*SW13=1        INHIBIT ERROR TYPEOUTS
8526                                    ;*SW10=1        BELL ON ERROR
8527                                    ;*SW09=1        LOOP ON ERROR
8528                                    ;*CALL
8529                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
8530
8531  043236                            $ERROR:
8532  043236  105237  001103            7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
8533  043242  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
8534  043244  013777  001102  135670            MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
8535  043252  032777  002000  135660            BIT     #BIT10,@SWR     ;;BELL ON ERROR?
8536  043260  001402                            BEQ     1$              ;;NO - SKIP
8537  043262  104401  001320                    TYPE    ,$BELL          ;;RING BELL
8538  043266  005237  001112            1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
8539  043272  011637  001116                    MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
8540  043276  162737  000002  001116            SUB     #2,$ERRPC
8541  043304  117737  135606  001114            MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
8542  043312  032777  020000  135620            BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
8543  043320  001004                            BNE     20$             ;;SKIP TYPEOUTS
8544  043322  004737  027506                    JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
8545  043326  104401  001325                    TYPE    ,$CRLF
8546  043332                            20$:
8547  043332  122737  000001  001350            CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
8548  043340  001007                            BNE     2$              ;;NO,SKIP APT ERROR REPORT
8549  043342  113737  001114  043354            MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
8550  043350  004737  044130                    JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
8551  043354     000                    21$:    .BYTE   0
8552  043355     000                            .BYTE   0
8553  043356  000777                    22$:    BR      22$             ;;APT ERROR LOOP
8554  043360  005777  135554            2$:     TST     @SWR            ;;HALT ON ERROR
8555  043364  100001                            BPL     3$              ;;SKIP IF CONTINUE
8556  043366  000000                            HALT                    ;;HALT ON ERROR!
8557  043370  032777  001000  135542    3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
8558  043376  001402                            BEQ     4$              ;;BR IF NO
8559  043400  013716  001110                    MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
8560  043404  005737  001316            4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
8561  043410  001402                            BEQ     5$              ;;BR IF NONE
8562  043412  013716  001316                    MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
8563  043416                            5$:
8564  043416  022737  016712  000042            CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
8565  043424  001001                            BNE     6$              ;;BRANCH IF NO
8566  043426  000000                            HALT                    ;;YES
8567  043430                            6$:
8568  043430  000002                            RTI                     ;;RETURN
8569                                    .SBTTL  TTY INPUT ROUTINE
8570
8571                                    ;;**************************************************************************
8572                                    .ENABL  LSB
8573
8574                                    .DSABL  LSB
8575
8576
```

```
8577                                    ;;*****************************************************************
8578                                    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8579                                    ;*CALL:
8580                                    ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
8581                                    ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
8582                                    ;*                              ;;WITH PARITY BIT STRIPPED OFF
8583                                    ;
8584
8585  043432  011646                    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
8586  043434  016666  000004  000002            MOV     4(SP),2(SP)     ;;SAVE THE PS
8587  043442  105777  135476           1$:     TSTB    @$TKS           ;;WAIT FOR
8588  043446  100375                            BPL     1$              ;;A CHARACTER
8589  043450  117766  135472  000004            MOVB    @$TKB,4(SP)     ;;READ THE TTY
8590  043456  042766  177600  000004            BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
8591  043464  026627  000004  000023            CMP     4(SP),#23       ;;IS IT A CONTROL-S?
8592  043472  001013                            BNE     3$              ;;BRANCH IF NO
8593  043474  105777  135444           2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
8594  043500  100375                            BPL     2$              ;;LOOP UNTIL ITS THERE
8595  043502  117746  135440                    MOVB    @$TKB,-(SP)     ;;GET CHARACTER
8596  043506  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
8597  043512  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
8598  043516  001366                            BNE     2$              ;;IF NOT DISCARD IT
8599  043520  000750                            BR      1$              ;;YES, RESUME
8600  043522  026627  000004  000140   3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
8601  043530  002407                            BLT     4$              ;;BRANCH IF YES
8602  043532  026627  000004  000175            CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
8603  043540  003003                            BGT     4$              ;;BRANCH IF YES
8604  043542  042766  000040  000004            BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
8605  043550  000002                    4$:     RTI                     ;;GO BACK TO USER
8606  043552  052536  005015     000   $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
8607  043557     136  006507  000012   $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
8608  043564  005015  053523  020122   $MSWR:  .ASCIZ  <15><12>/SWR = /
8609  043572  020075     000
8610  043575     040  047040  053505   $MNEW:  .ASCIZ  /  NEW = /
8611  043602  036440  000040
8612
8613
8614
8615                                    .SBTTL  POWER DOWN AND UP ROUTINES
8616
8617                                    ;POWER DOWN ROUTINE
8618  043606  012737  043620  000024  $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
8619  043614  000000                            HALT                    ;HANG UP
8620  043616  000776                            BR      .-2
8621
8622                                    ;POWER UP ROUTINE
8623  043620  005037  043672           $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
8624  043624  005237  043672           4$:     INC     $PWRCT
8625  043630  001375                            BNE     4$
8626  043632  012737  043606  000024            MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
8627  043640  012737  000340  000026            MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
8628  043646  012737  000340  000036            MOV     #PR7,TRAPVEC+2
8629  043654  012706  001100                    MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
8630  043660  104401  043674                    TYPE    ,PWRMSG         ;TYPE "POWER FAILED"
8631  043664  000005                            RESET                   ;CLEAR THE UNIBUS
8632  043666  000177  135214                    JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
 8633  043672  000000                    $PWRCT: .WORD   0
 8634  043674  005015  047520  042527    PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
 8635  043702  020122  040506  046111
 8636  043710  042105  005015     000
 8637          043716                            .EVEN
 8638
 8639
 8640
 8641                                    ;;**********************************************************************
 8642                                    ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
 8643                                    ;*CALLED BY "SCOPER"
 8644                                    ;;**********************************************************************
 8645  043716  105737  001103           SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
 8646  043722  001406                           BEQ     6$              ;BR IF NOT
 8647  043724  032777  001000  135206           BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
 8648  043732  001402                           BEQ     6$              ;BR IF NOT
 8649  043734  013716  001110                   MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
 8650  043740  000002                   6$:     RTI                     ;RETURN
 8651
 8652
 8653
 8654                                    .SBTTL  SCOPE HANDLER ROUTINE
 8655
 8656                                    ;;**********************************************************************
 8657                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 8658                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 8659                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 8660                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 8661                                    ;*SW14=1         LOOP ON TEST
 8662                                    ;*SW09=1         LOOP ON ERROR
 8663                                    ;*CALL
 8664                                    ;*      SCOPE           ;;SCOPE=IOT
 8665
 8666  043742                            $SCOPE:
 8667  043742  032777  040000  135170    1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 8668  043750  001052                           BNE     $OVER           ;;YES IF SW14=1
 8669                                    ;#####START OF CODE FOR THE XOR TESTER#####
 8670  043752  000416                   $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
 8671                                                                    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 8672  043754  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 8673  043760  012737  044000  000004           MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 8674  043766  005737  177060                   TST     @#177060        ;;TIME OUT ON XOR?
 8675  043772  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 8676  043776  000421                           BR      $SVLAD          ;;GO TO THE NEXT TEST
 8677  044000  022626                   5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 8678  044002  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 8679  044006  000407                           BR      7$              ;;LOOP ON THE PRESENT TEST
 8680  044010                            6$:;#####END OF CODE FOR THE XOR TESTER#####
 8681  044010  105737  001103           2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 8682  044014  001412                           BEQ     $SVLAD          ;;BR IF NO
 8683  044016  032777  001000  135114           BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 8684  044024  001404                           BEQ     4$              ;;BR IF NO
 8685  044026  013737  001110  001106   7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 8686  044034  000420                           BR      $OVER
 8687  044036  105037  001103           4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 8688  044042  105237  001102           $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
 8689  044046  113737  001102  001334          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 8690  044054  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 8691  044060  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 8692  044064  005037  001316                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 8693  044070  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 8694  044076  013777  001102  135036  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 8695  044104  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 8696  044110  000002                          RTI                     ;;FIXES PS
 8697                                  .SBTTL  APT COMMUNICATIONS ROUTINE
 8698
 8699                                  ;;**************************************************************************
 8700  044112  112737  000001  044356  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
 8701  044120  112737  000001  044354  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
 8702  044126  000403                          BR      $ATYC
 8703  044130  112737  000001  044356  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
 8704  044136                          $ATYC:
 8705  044136  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 8706  044140  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 8707  044142  105737  044354                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
 8708  044146  001450                          BEQ     5$              ;;IF NOT:  BR
 8709  044150  122737  000001  001350          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
 8710  044156  001031                          BNE     3$              ;;IF NOT:  BR
 8711  044160  132737  000100  001351          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 8712  044166  001425                          BEQ     3$              ;;IF NOT:  BR
 8713  044170  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
 8714  044174  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 8715  044202  005737  001330          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
 8716  044206  001375                          BNE     1$              ;;IF NOT:  WAIT
 8717  044210  010037  001344                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
 8718  044214  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
 8719  044216  001376                          BNE     2$
 8720  044220  163700  001344                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
 8721  044224  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
 8722  044226  010037  001346                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
 8723  044232  012737  000004  001330          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
 8724  044240  000413                          BR      5$
 8725  044242  017637  000004  044266  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
 8726  044250  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 8727  044256  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 8728  044262  004737  041510                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
 8729  044266  000000                  4$:     .WORD   0
 8730  044270                          5$:
 8731  044270  105737  044356          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 8732  044274  001416                          BEQ     12$             ;;IF NOT:  BR
 8733  044276  005737  001350                  TST     $ENV            ;;RUNNING UNDER APT?
 8734  044302  001413                          BEQ     12$             ;;IF NOT:  BR
 8735  044304  005737  001330          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 8736  044310  001375                          BNE     11$             ;;IF NOT:  WAIT
 8737  044312  017637  000004  001332          MOV     @4(SP),$FATAL   ;;GET ERROR #
 8738  044320  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 8739  044326  005237  001330                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 8740  044332  105037  044356          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 8741  044336  105037  044355                  CLRB    $LFLG           ;;CLEAR LOG FLAG
 8742  044342  105037  044354                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 8743  044346  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 8744  044350  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
8745  044352  000207                  RTS     PC              ;;RETURN
8746  044354     000                  $MFLG:  .BYTE   0       ;;MESSG. FLAG
8747  044355     000                  $LFLG:  .BYTE   0       ;;LOG FLAG
8748  044356     000                  $FFLG:  .BYTE   0       ;;FATAL FLAG
8749          044360                          .EVEN
8750          000200                  APTSIZE=200
8751          000001                  APTENV=001
8752          000100                  APTSPOOL=100
8753          000040                  APTCSUP=040
8754                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
8755
8756                                  ;;**********************************************************************
8757                                  ;*SAVE R0-R5
8758                                  ;*CALL:
8759                                  ;*      SAVREG
8760                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
8761                                  ;*
8762                                  ;*TOP---(+16)
8763                                  ;* +2---(+18)
8764                                  ;* +4---R5
8765                                  ;* +6---R4
8766                                  ;* +8---R3
8767                                  ;*+10---R2
8768                                  ;*+12---R1
8769                                  ;*+14---R0
8770
8771  044360                          $SAVREG:
8772  044360  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8773  044362  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8774  044364  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8775  044366  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8776  044370  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
8777  044372  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
8778  044374  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
8779  044400  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
8780  044404  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
8781  044410  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
8782  044414  000002                          RTI
8783
8784                                  ;*RESTORE R0-R5
8785                                  ;*CALL:
8786                                  ;*      RESREG
8787  044416                          $RESREG:
8788  044416  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
8789  044422  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
8790  044426  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
8791  044432  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
8792  044436  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
8793  044440  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
8794  044442  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8795  044444  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8796  044446  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8797  044450  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8798  044452  000002                          RTI
8799                                  .SBTTL  TRAP DECODER
8800
```

```
8801                                    ;;**********************************************************************
8802                                    ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8803                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8804                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8805                                    ;*GO TO THAT ROUTINE.
8806
8807  044454  010046                    $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
8808  044456  016600  000002                   MOV     2(SP),R0        ;;GET TRAP ADDRESS
8809  044462  005740                           TST     -(R0)           ;;BACKUP BY 2
8810  044464  111000                           MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
8811  044466  006300                           ASL     R0              ;;POSITION FOR INDEXING
8812  044470  016000  044510                   MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
8813  044474  000200                           RTS     R0              ;;GO TO ROUTINE
8814
8815
8816                                    ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8817
8818  044476  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
8819  044500  016666  000004  000002            MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
8820  044506  000002                           RTI                     ;;RESTORE THE PSW
8821
8822                                    .SBTTL  TRAP TABLE
8823
8824                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8825                                    ;*BY THE "TRAP" INSTRUCTION.
8826
8827                                    ;       ROUTINE
8828                                    ;       -------
8829  044510  044476                    $TRPAD: .WORD   $TRAP2
8830  044512  041510                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
8831  044514  042610                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
8832  044516  042564                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
8833  044520  042624                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
8834  044522  043012                            $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
8835
8836
8837  044524  043432                            $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
8838  044526  044360                            $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
8839  044530  044416                            $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
8840  044532  043716                            SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
8841
8842  044534  020040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
8843  044542  052123  000040
8844  044546  025052  020040     000  AS2SP2: .ASCIZ  /**  /
8845  044553     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PARITY ERROR/
8846  044560  020123  040520  044522
8847  044566  054524  042440  051122
8848  044574  051117     000
8849  044577     116  047117  042455  EM2:    .ASCIZ  /NON-EXISTANT MEMORY/
8850  044604  044530  052123  047101
8851  044612  020124  042515  047515
8852  044620  054522     000
8853  044623     116  047117  042455  EM3:    .ASCIZ  /NON-EXISTANT DRIVE/
8854  044630  044530  052123  047101
8855  044636  020124  051104  053111
8856  044644  000105
```

```
8857  044646  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERROR/
8858  044654  042511  042114  042440
8859  044662  051122  051117     000
8860  044667     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
8861  044674  020123  044524  042515
8862  044702  052517  000124
8863  044706  020104  047524  041440  EM6:    .ASCIZ  /D TO C PARITY ERROR/
8864  044714  050040  051101  052111
8865  044722  020131  051105  047522
8866  044730  000122
8867  044732  051104  053111  020105  EM7:    .ASCIZ  /DRIVE DETECTED PARITY ERROR/
8868  044740  042504  042524  052103
8869  044746  042105  050040  051101
8870  044754  052111  020131  051105
8871  044762  047522  000122
8872  044766  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
8873  044774     000
8874  044775     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
8875  045002  046040  051517  000123
8876  045010  046111  042514  040507  EM12:   .ASCIZ  /ILLEGAL FUNCTION/
8877  045016  020114  052506  041516
8878  045024  044524  047117     000
8879  045031     120  047522  051107  EM13:   .ASCIZ  /PROGRAMMING ERROR/
8880  045036  046501  044515  043516
8881  045044  042440  051122  051117
8882  045052     000
8883  045053     116  047117  042455  EM14:   .ASCIZ  /NON-EXISTANT FUNCTION/
8884  045060  044530  052123  047101
8885  045066  020124  052506  041516
8886  045074  044524  047117     000
8887  045101     104  044522  042526  EM15:   .ASCIZ  /DRIVE TYPE ERROR/
8888  045106  052040  050131  020105
8889  045114  051105  047522  000122
8890  045122  047506  046522  052101  EM16:   .ASCIZ  /FORMAT ERROR/
8891  045130  042440  051122  051117
8892  045136     000
8893  045137     127  044522  042524  EM17:   .ASCIZ  /WRITE LOCK ERROR/
8894  045144  046040  041517  020113
8895  045152  051105  047522  000122
8896  045160  051104  053111  020105  EM20:   .ASCIZ  /DRIVE UNSAFE/
8897  045166  047125  040523  042506
8898  045174     000
8899  045175     123  042505  020113  EM21:   .ASCIZ  /SEEK INCOMPLETE/
8900  045202  047111  047503  050115
8901  045210  042514  042524     000
8902  045215     103  046131  047111  EM22:   .ASCIZ  /CYLINDER OVERFLOW/
8903  045222  042504  020122  053117
8904  045230  051105  046106  053517
8905  045236     000
8906  045237     111  046114  043505  EM23:   .ASCIZ  /ILLEGAL CYLINDER ADDRESS/
8907  045244  046101  041440  046131
8908  045252  047111  042504  020122
8909  045260  042101  051104  051505
8910  045266  000123
8911  045270  051104  053111  020105  EM24:   .ASCIZ  /DRIVE OFF TRACK/
8912  045276  043117  020106  051124
```

```
8913  045304  041501  000113
8914  045310  051104  053111  020105  EM25:   .ASCIZ  /DRIVE TIMING ERROR/
8915  045316  044524  044515  043516
8916  045324  042440  051122  051117
8917  045332      000
8918  045333      104  052101  020101  EM26:   .ASCIZ  /DATA LATE/
8919  045340  040514  042524      000
8920  045345      103  047117  051124  EM27:   .ASCIZ  /CONTROLLER TIMEOUT/
8921  045352  046117  042514  020122
8922  045360  044524  042515  052517
8923  045366  000124
8924  045370  050117  051105  052101  EM30:   .ASCIZ  /OPERATION INCOMPLETE/
8925  045376  047511  020116  047111
8926  045404  047503  050115  042514
8927  045412  042524      000
8928  045415      110  040505  042504  EM31:   .ASCIZ  /HEADER VRC ERROR/
8929  045422  020122  051126  020103
8930  045430  051105  047522  000122
8931  045436  040504  040524  041440  EM32:   .ASCIZ  /DATA CHECK ERROR/
8932  045444  042510  045503  042440
8933  045452  051122  051117      000
8934  045457      127  044522  042524  EM33:   .ASCIZ  /WRITE CHECK ERROR/
8935  045464  041440  042510  045503
8936  045472  042440  051122  051117
8937  045500      000
8938  045501      104  052101  020101  EM34:   .ASCIZ  /DATA MISCOMPARE/
8939  045506  044515  041523  046517
8940  045514  040520  042522      000
8941  045521      116  020117  051104  EM35:   .ASCIZ  /NO DRIVE RESPONSE-UFE & NXD/
8942  045526  053111  020105  042522
8943  045534  050123  047117  042523
8944  045542  052455  042506  023040
8945  045550  047040  042130      000
8946  045555      104  044522  042526  EM36:   .ASCIZ  /DRIVE ERROR WILL NOT CLEAR/
8947  045562  042440  051122  051117
8948  045570  053440  046111  020114
8949  045576  047516  020124  046103
8950  045604  040505  000122
8951  045610  051104  053111  020105  EM37:   .ASCIZ  /DRIVE STATUS CHANGE WILL NOT CLEAR/
8952  045616  052123  052101  051525
8953  045624  041440  040510  043516
8954  045632  020105  044527  046114
8955  045640  047040  052117  041440
8956  045646  042514  051101      000
8957  045653      101  052124  047047  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
8958  045660  041040  052125  047040
8959  045666  020117  052123  052101
8960  045674  051525  041440  040510
8961  045702  043516  020105  051117
8962  045710  043040  052501  052114
8963  045716      000
8964  045717      101  052124  047047  EM41:   .ASCIZ  /ATT'N BUT DRIVE NOT AVAILABLE/
8965  045724  041040  052125  042040
8966  045732  044522  042526  047040
8967  045740  052117  040440  040526
8968  045746  046111  041101  042514
```

```
8969  045754      000
8970  045755      101  052124  047047  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
8971  045762  053440  042510  020116
8972  045770  047516  020124  054105
8973  045776  042520  052103  042105
8974  046004      000
8975  046005      105  051122  051117  EM43:   .ASCIZ  /ERROR GATHERING DRIVE STATUS/
8976  046012  043440  052101  042510
8977  046020  044522  043516  042040
8978  046026  044522  042526  051440
8979  046034  040524  052524  000123
8980  046042  052515  052114  050111  EM52:   .ASCIZ  /MULTIPLE DRIVE SELECT/
8981  046050  042514  042040  044522
8982  046056  042526  051440  046105
8983  046064  041505  000124
8984  046070  042510  042101  051105  EM53:   .ASCIZ  /HEADER COMPARE ERROR/
8985  046076  041440  046517  040520
8986  046104  042522  042440  051122
8987  046112  051117      000
8988  046115      123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
8989  046122  020123  044524  042515
8990  046130  052517  000124
8991  046134  051105  047522  020122  EM60:   .ASCIZ  /ERROR IN RECAL FOR RECOVERY/
8992  046142  047111  051040  041505
8993  046150  046101  043040  051117
8994  046156  051040  041505  053117
8995  046164  051105  000131
8996  046170  051120  043517  040522  EM61:   .ASCIZ  /PROGRAM ABORTING FATAL ERROR IN RETRY/
8997  046176  020115  041101  051117
8998  046204  044524  043516  043040
8999  046212  052101  046101  042440
9000  046220  051122  051117  044440
9001  046226  020116  042522  051124
9002  046234  000131
9003  046236  054503  044514  042116  EM62:   .ASCIZ  /CYLINDER MISCOMPARE/
9004  046244  051105  046440  051511
9005  046252  047503  050115  051101
9006  046260  000105
9007  046262  046103  040505  020122  EM63:   .ASCIZ  /CLEAR CONTROLLER DID NOT CLEAR ERROR/
9008  046270  047503  052116  047522
9009  046276  046114  051105  042040
9010  046304  042111  047040  052117
9011  046312  041440  042514  051101
9012  046320  042440  051122  051117
9013  046326      000
9014  046327      116  020117  052101  EM64:   .ASCIZ  /NO ATT'N IN ATT'N SUMMARY REGISTER/
9015  046334  023524  020116  047111
9016  046342  040440  052124  047047
9017  046350  051440  046525  040515
9018  046356  054522  051040  043505
9019  046364  051511  042524  000122
9020  046372  047125  047523  044514  EM65:   .ASCIZ  /UNSOLICITED ATTENTION/
9021  046400  044503  042524  020104
9022  046406  052101  042524  052116
9023  046414  047511  000116
9024  046420  047125  054105  042520  EM66:   .ASCIZ  /UNEXPECTED DATA TYPE ERROR/
```

J14


```
9025  046426  052103  042105  042040
9026  046434  052101  020101  054524
9027  046442  042520  042440  051122
9028  046450  051117     000
9029  046453     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
9030  046460  042040  042111  047040
9031  046466  052117  051040  051505
9032  046474  052105  053440  052111
9033  046502  020110  046103  040505
9034  046510  000122
9035  046512  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
9036  046520  041440  042514  051101
9037  046526  042040  042111  047040
9038  046534  052117  041440  042514
9039  046542  051101  042040  044522
9040  046550  042526  040440  052124
9041  046556  047047     000
9042  046561     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/
9043  046566  040514  042524  053440
9044  046574  042510  020116  047125
9045  046602  047514  042101  047111
9046  046610  020107  042510  042101
9047  046616  051105     000
9048  046621     103  047117  051124  EM72:   .ASCIZ  /CONTROLLER ERROR WHEN DRIVER SERVICING/
9049  046626  046117  042514  020122
9050  046634  051105  047522  020122
9051  046642  044127  047105  042040
9052  046650  044522  042526  020122
9053  046656  042523  053122  041511
9054  046664  047111  000107
9055  046670  051104  053111  020105  EM73:   .ASCIZ  /DRIVE DETECTED PARITY ERROR/
9056  046676  042504  042524  052103
9057  046704  042105  050040  051101
9058  046712  052111  020131  051105
9059  046720  047522  000122
9060  046724  047125  042504  044506  EM74:   .ASCIZ  /UNDEFINED ERROR/
9061  046732  042516  020104  051105
9062  046740  047522  000122
9063  046744  040515  045522  047111  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
9064  046752  020107  044124  051511
9065  046760  051440  041505  047524
9066  046766  020122  040502  000104
9067  046774  040502  020104  040504  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
9068  047002  040524  053040  051105
9069  047010  043111  047047  053440
9070  047016  052111  020110  042522
9071  047024  042101  020056  041505
9072  047032  020103  043117  046040
9073  047040  051501  020124  042522
9074  047046  051124  020131  051511
9075  047054  000072
9076  047056  042522  051124  020131  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
9077  047064  052523  041503  051505
9078  047072  043123  046125     000
9079  047077     122  052105  054522  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
9080  047104  052440  051516  041525
```

K14


```
9081  047112  042503  051523  052506
9082  047120  000114
9083  047122  040503  047116  052117  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
9084  047130  043040  047111  020104
9085  047136  020101  040526  044514
9086  047144  020104  042510  042101
9087  047152  051105  044440  020116
9088  047160  051124  041501  020113
9089  047166  052512  052123  051040
9090  047174  040505  000104
9091  047200  040502  020104  042523  EM102:  .ASCIZ  /BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
9092  047206  052103  051117  042440
9093  047214  051122  051117  047440
9094  047222  020116  042523  052103
9095  047230  051117  047040  052117
9096  047236  046040  051511  042524
9097  047244  020104  040502  000104
9098  047252  044524  042515  026504  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
9099  047260  052517  020124  047117
9100  047266  051040  040505  020104
9101  047274  042110  000122
9102  047300  044524  042515  026504  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
9103  047306  052517  020124  047117
9104  047314  051440  042505  000113
9105  047322  051104  053111  020105  EM105:  .ASCIZ  /DRIVE SIEZED BY OTHER PORT/
9106  047330  044523  055105  042105
9107  047336  041040  020131  052117
9108  047344  042510  020122  047520
9109  047352  052122  000
9110  047355  000104  052101  020101  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
9111  047362  044515  041523  050115
9112  047370  020122  044127  046111
9113  047376  020105  040502  020111
9114  047404  042523  000124
9115  047410  047516  047040  046505  EM107:  .ASCIZ  /NO NEM ERROR WHEN REF'ING LOC 760000/
9116  047416  042440  051122  051117
9117  047424  053440  042510  020116
9118  047432  042522  023506  047111
9119  047440  020107  047514  020103
9120  047446  033067  030060  030060
9121  047454  000000
9122  047455  000111  052116  050122  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
9123  047462  020124  044127  047105
9124  047470  041440  052116  046122
9125  047476  020122  047516  020124
9126  047504  042122  000131
9127  047510  047516  040440  052124  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
9128  047516  047047  047440  020116
9129  047524  042523  045505  000000
9130  047531  000104  044522  042526  EM112:  .ASCIZ  /DRIVE'S CYLINDER INCORRECT/
9131  047536  051447  041440  046131
9132  047544  047111  042504  020122
9133  047552  047111  047503  051122
9134  047560  041505  000124
9135  047564  041101  051117  026524  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
9136  047572  041440  047101  052047
```

```
 9137  047600  051040  040505  020104
 9138  047606  051502  000106
 9139  047612  052113  030461  043040  EM114:  .ASCIZ  /KT11 FAILURE/
 9140  047620  044501  052514  042522
 9141  047626     000
 9142  047627     115  046505  050040  EM115:  .ASCIZ  /MEM PARITY ERR/
 9143  047634  051101  052111  020131
 9144  047642  051105  000122
 9145  047646  052503  051122  047105  DH100:  .ASCIZ  /CURRENT COMMAND :/
 9146  047654  020124  047503  046515
 9147  047662  047101  020104  000072
 9148  047670  051120  053105  047511  DH105:  .ASCIZ  /PREVIOUS COMMAND :/
 9149  047676  051525  041440  046517
 9150  047704  040515  042116  035040
 9151  047712     000
 9152  047713     122  033113  030461  DH200:  .ASCIZ  /RK611 REGISTERS :/
 9153  047720  051040  043505  051511
 9154  047726  042524  051522  035040
 9155  047734     000
 9156  047735     122  046505  044501  DH500:  .ASCIZ  /REMAINING REGISTERS NOT VALID/
 9157  047742  044516  043516  051040
 9158  047750  043505  051511  042524
 9159  047756  051522  047040  052117
 9160  047764  053040  046101  042111
 9161  047772     000
 9162  047773     124  042510  043040  DH501:  .ASCIZ  /THE FOLLOWING REGISTER DATA MAY BE INCORRECT :/
 9163  050000  046117  047514  044527
 9164  050006  043516  051040  043505
 9165  050014  051511  042524  020122
 9166  050022  040504  040524  046440
 9167  050030  054501  041040  020105
 9168  050036  047111  047503  051122
 9169  050044  041505  020124  000072
 9170  050052  051105  020122  041520  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT/
 9171  050060  020040  051104  053111
 9172  050066  020105  020040  046503
 9173  050074  042116  020040  020040
 9174  050102  054503  047114  051104
 9175  050110  020040  051124  041501
 9176  050116  020113  020040  042523
 9177  050124  052103  051117  020040
 9178  050132  042127  041440  052116
 9179  050140     000
 9180  050141     110  020111  040502  DH102:  .ASCIZ  /HI BA   LO BA/
 9181  050146  020040  046040  020117
 9182  050154  040502     000
 9183  050157     120  042522  027126  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
 9184  050164  052440  044516  052502
 9185  050172  020123  040515  020120
 9186  050200  000072
 9187  050202  044510  051040  043505  DH104:  .ASCIZ  /HI REGO LO REGO/
 9188  050210  020060  047514  051040
 9189  050216  043505  000060
 9190  050222  052503  027122  052440  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
 9191  050230  044516  052502  020123
 9192  050236  040515  020120  000072
```

```
9193  050244  045522  051503  020061  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
9194  050252  020040  045522  051503
9195  050260  020062  020040  045522
9196  050266  041504  020040  020040
9197  050274  045522  040504  020040
9198  050302  020040  045522  041527
9199  050310  020122  020040  045522
9200  050316  040502  020040  020040
9201  050324  045522  051501  043117
9202  050332      000
9203  050333      122  042113  020123  DH202:  .ASCIZ  /RKDS    RKER/
9204  050340  020040  051040  042513
9205  050346  000122
9206  050350  030101  020060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
9207  050356  020040  030102  020060
9208  050364  020040  020040  030101
9209  050372  020061  020040  020040
9210  050400  030102  020061  020040
9211  050406  020040  030101  020062
9212  050414  020040  020040  030102
9213  050422  020062  020040  020040
9214  050430  030101  020063  020040
9215  050436  020040  030102  000063
9216  050444  042510  042101  051105  DH601:  .ASCIZ  /HEADER SHOULD BE:/
9217  050452  051440  047510  046125
9218  050460  020104  042502  000072
9219  050466  042510  042101  051105  DH602:  .ASCIZ  /HEADER OF SECTOR 0 THIS CYLINDER IS:/
9220  050474  047440  020106  042523
9221  050502  052103  051117  030040
9222  050510  052040  044510  020123
9223  050516  054503  044514  042116
9224  050524  051105  044440  035123
9225  050532      000
9226  050533      120  041501  020113  DH604:  .ASCIZ  /PACK ADDRESS OF ERROR(S) :/
9227  050540  042101  051104  051505
9228  050546  020123  043117  042440
9229  050554  051122  051117  051450
9230  050562  020051  000072
9231  050566  054503  047114  051104  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
9232  050574  020040  051124  041501
9233  050602  020113  020040  042523
9234  050610  052103  051117      000
9235  050615      103  046131  042116  DH6042: .ASCIZ  /CYLNDR  TRACK/
9236  050622  020122  052040  040522
9237  050630  045503      000
9238  050633      105  041503  042040  DH605:  .ASCIZ  /ECC DATA IS:/
9239  050640  052101  020101  051511
9240  050646  000072
9241  050650  047520  020123  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
9242  050656  020040  040520  020124
9243  050664  020040  020040  047503
9244  050672  051122  041505  040524
9245  050700  046102  037505      000
9246  050705      127  051117  030504  DH606:  .ASCII  /WORD1   WORD2   WORD3/
9247  050712  020040  053440  051117
9248  050720  031104  020040  053440
```

```
 9249  050726  051117  031504     000
 9250  050733     102  042101  044040  DH607:  .ASCIZ  /BAD HEADER IS :/
 9251  050740  040505  042504  020122
 9252  050746  051511  035040     000
 9253  050753     127  020104  020043  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
 9254  050760  020040  043440  047517
 9255  050766  020104  020040  041040
 9256  050774  042101  020040  020040
 9257  051002  044040  020111  044120
 9258  051010  020131  046040  020117
 9259  051016  044120  020131  053040
 9260  051024  052122  040440  020104
 9261  051032  045440  050111  051101
 9262  051040  000066
 9263  051042  047507  042117  020040  DH702:  .ASCIZ  /GOOD    BAD/
 9264  051050  020040  040502  000104
 9265  051056  040506  046111  047111  DH703:  .ASCIZ  /FAILING DATA WORD :/
 9266  051064  020107  040504  040524
 9267  051072  053440  051117  020104
 9268  051100  000072
 9269  051102  051123  004460  051123  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
 9270  051110  004461  051123  004462
 9271  051116  051123  000063
 9272  051122  052113  030461  051040  DH705:  .ASCIZ  /KT11 REGS :/
 9273  051130  043505  020123  000072
 9274  051136  042515  020115  042522  DH706:  .ASCIZ  /MEM REGS :/
 9275  051144  051507  035040     000
 9276  051151     120  004503  047514  DH707:  .ASCIZ  /PC      LOERAD  HIERAD  MEMSYS/
 9277  051156  051105  042101  044011
 9278  051164  042511  040522  004504
 9279  051172  042515  051515  051531
 9280  051200     000
 9281  051201     120  004503  051503  DH710:  .ASCIZ  /PC      CSR AD  CSR/
 9282  051206  020122  042101  041411
 9283  051214  051123     000
 9284  051217     103  046131  047111  DH711:  .ASCIZ  /CYLINDERS :/
 9285  051224  042504  051522  035040
 9286  051232     000
 9287  051233     122  042113  020123  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
 9288  051240  020040  051040  042513
 9289  051246  020122  020040  051040
 9290  051254  046513  031122  020040
 9291  051262  051040  046513  031522
 9292  051270     000
 9293  051271     105  051122  051117  DH502:  .ASCIZ  /ERROR TRYING TO GET FAILURE INFO/
 9294  051276  052040  054522  047111
 9295  051304  020107  047524  043440
 9296  051312  052105  043040  044501
 9297  051320  052514  042522  044440
 9298  051326  043116  000117
 9299  051332  042523  047503  042116  DH503:  .ASCIZ  /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
 9300  051340  052040  046511  047505
 9301  051346  052125  047440  041503
 9302  051354  051125  042522  020104
 9303  051362  042507  052124  047111
 9304  051370  020107  051104  053111
```

```
 9305  051376  020105  052123  052101
 9306  051404  051525     000
 9307  051407     116  046525  042502  DH800:  .ASCIZ  /NUMBER OF RETRIES:/
 9308  051414  020122  043117  051040
 9309  051422  052105  044522  051505
 9310  051430  000072
 9311                                  .EVEN
 9312  051432  001116  005420  005416  DT100:  .WORD   $ERRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
 9313  051440  001162  001164  001166
 9314  051446  001170
 9315  051450  001256  001260          DT102:  .WORD   $REG36,$REG37
 9316  051454  001174  001176  001200  DT201:  .WORD   $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
 9317  051462  001202  001204  001206
 9318  051470  001210
 9319  051472  001212  001214          DT202:  .WORD   $REG14,$REG15
 9320  051476  001216  001220  001222  DT203:  .WORD   $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
 9321  051504  001224  001226  001230
 9322  051512  001232  001234
 9323  051516  001174  001176  001200  DT601:  .WORD   $REG5,$REG6,$REG7
 9324  051524  001202  001204  001206  DT602:  .WORD   $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
 9325  051532  001210  001212  001214
 9326  051540  001216
 9327
 9328  051542  000006                  DF01:   .WORD   6               ;NUMBER OF HEADER LINES
 9329  051544     000                          .BYTE   0               ;NUMBER OF COL FOR FIRST HDR
 9330  051545     000                          .BYTE   0               ;ALL CHARACTERS OCTAL
 9331  051546  050052                          .WORD   DH101           ;SECOND HDR LINE
 9332  051550     007     000                  .BYTE   7,0             ;NUM OF COL-ALL OCTAL
 9333  051552  050141                          .WORD   DH102
 9334  051554     002     000                  .BYTE   2,0
 9335  051556  047713                          .WORD   DH200
 9336  051560     000     000                  .BYTE   0,0
 9337  051562  050244                          .WORD   DH201
 9338  051564     007     000                  .BYTE   7,0
 9339  051566  047735                          .WORD   DH500
 9340  051570     000     000                  .BYTE   0,0
 9341
 9342  051572  000007                  DF02:   .WORD   7
 9343  051574     000     000                  .BYTE   0,0
 9344  051576  050052                          .WORD   DH101
 9345  051600     007     000                  .BYTE   7,0
 9346  051602  050141                          .WORD   DH102
 9347  051604     002     000                  .BYTE   2,0
 9348  051606  047713                          .WORD   DH200
 9349  051610     000     000                  .BYTE   0,0
 9350  051612  050244                          .WORD   DH201
 9351  051614     007     000                  .BYTE   7,0
 9352  051616  050333                          .WORD   DH202
 9353  051620     002     000                  .BYTE   2,0
 9354  051622  050350                          .WORD   DH203
 9355  051624     010     000                  .BYTE   10,0
 9356
 9357  051626  000010                  DF03:   .WORD   10
 9358  051630     000     000                  .BYTE   0,0
 9359  051632  050052                          .WORD   DH101
 9360  051634     007     000                  .BYTE   7,0
```

```
9361  051636  050141                          .WORD   DH102
9362  051640     002      000                 .BYTE   2,0
9363  051642  047713                          .WORD   DH200
9364  051644     000      000                 .BYTE   0,0
9365  051646  050244                          .WORD   DH201
9366  051650     007      000                 .BYTE   7,0
9367  051652  047773                          .WORD   DH501          ;"THE FOLLOWING REG DATA MB INCORRECT"
9368  051654     000      000                 .BYTE   0,0
9369  051656  050333                          .WORD   DH202
9370  051660     002      000                 .BYTE   2,0
9371  051662  050350                          .WORD   DH203
9372  051664     010      000                 .BYTE   10,0
9373
9374  051666  000003                  DF04:   .WORD   3
9375  051670     000      000                 .BYTE   0,0
9376  051672  050052                          .WORD   DH101
9377  051674     007      000                 .BYTE   7,0
9378  051676  050141                          .WORD   DH102
9379  051700     002      000                 .BYTE   2,0
9380
9381  051702  000007                  DF05:   .WORD   7              ;OPI, HVRC
9382  051704     000      000                 .BYTE   0,0
9383  051706  050052                          .WORD   DH101
9384  051710     007      000                 .BYTE   7,0
9385  051712  050141                          .WORD   DH102
9386  051714     002      000                 .BYTE   2,0
9387  051716  050444                          .WORD   DH601
9388  051720     000      000                 .BYTE   0,0
9389  051722  050705                          .WORD   DH606
9390  051724     003      000                 .BYTE   3,0
9391  051726  050466                          .WORD   DH602
9392  051730     000      000                 .BYTE   0,0
9393  051732  050705                          .WORD   DH606
9394  051734     003      000                 .BYTE   3,0
9395
9396  051736  000007                  DF07:   .WORD   7
9397  051740     000      000                 .BYTE   0,0
9398  051742  050052                          .WORD   DH101
9399  051744     007      000                 .BYTE   7,0
9400  051746  050141                          .WORD   DH102
9401  051750     002      000                 .BYTE   2,0
9402  051752  050533                          .WORD   DH604
9403  051754     000      000                 .BYTE   0,0
9404  051756  050566                          .WORD   DH6041
9405  051760     003      000                 .BYTE   3,0
9406  051762  050633                          .WORD   DH605
9407  051764     000      000                 .BYTE   0,0
9408  051766  050650                          .WORD   DH6051
9409  051770     003      000                 .BYTE   3,0
9410
9411  051772  000005                  DF10:   .WORD   5
9412  051774     000      000                 .BYTE   0,0
9413  051776  050052                          .WORD   DH101
9414  052000     007      000                 .BYTE   7,0
9415  052002  050141                          .WORD   DH102
9416  052004     002      000                 .BYTE   2,0
```

```
9417  052006  050533                    .WORD   DH604
9418  052010     000      000           .BYTE   0,0
9419  052012  050566                    .WORD   DH6041
9420  052014     003      000           .BYTE   3,0
9421
9422  052016  000004              DF11: .WORD   4
9423  052020     000      000           .BYTE   0,0
9424  052022  050533                    .WORD   DH604
9425  052024     000      000           .BYTE   0,0
9426  052026  050566                    .WORD   DH6041
9427  052030     003      000           .BYTE   3,0
9428  052032  050753                    .WORD   DH701
9429  052034     000      000           .BYTE   0,0
9430
9431  052036  000006              DF12: .WORD   6
9432  052040     000      000           .BYTE   0,0
9433  052042  050052                    .WORD   DH101
9434  052044     007      000           .BYTE   7,0
9435  052046  050141                    .WORD   DH102
9436  052050     002      000           .BYTE   2,0
9437  052052  047713                    .WORD   DH200
9438  052054     000      000           .BYTE   0,0
9439  052056  050244                    .WORD   DH201
9440  052060     007      000           .BYTE   7,0
9441  052062  051233                    .WORD   DH204
9442  052064     004      000           .BYTE   4,0
9443
9444  052066  000006              DF13: .WORD   6           ;FORMAT FOR 2ND LEVEL ERROR
9445  052070     000      000           .BYTE   0,0         ;IN HEADER COMPARE ERROR
9446  052072  050052                    .WORD   DH101       ;AND 2ND LEVEL HEADER
9447  052074     007      000           .BYTE   7,0         ;VRC ERROR
9448  052076  050141                    .WORD   DH102
9449  052100     002      000           .BYTE   2,0
9450  052102  050444                    .WORD   DH601
9451  052104     000      000           .BYTE   0,0
9452  052106  050705                    .WORD   DH606
9453  052110     003      000           .BYTE   3,0
9454  052112  051271                    .WORD   DH502
9455  052114     000      000           .BYTE   0,0
9456
9457  052116  000006              DF14: .WORD   6           ;FORMAT FOR 2ND LEVEL ERROR
9458  052120     000      000           .BYTE   0,0         ;IN OPERATION INCOMPLETE ERROR
9459  052122  050052                    .WORD   DH101
9460  052124     007      000           .BYTE   7,0
9461  052126  050141                    .WORD   DH102
9462  052130     002      000           .BYTE   2,0
9463  052132  050444                    .WORD   DH601
9464  052134     000      000           .BYTE   0,0
9465  052136  050705                    .WORD   DH606
9466  052140     003      000           .BYTE   3,0
9467  052142  051271                    .WORD   DH502
9468  052144     000      000           .BYTE   0,0
9469
9470  052146  000011              DF15: .WORD   11
9471  052150     000      000           .BYTE   0,0
9472  052152  050052                    .WORD   DH101
```

```
9473  052154  007     000             .BYTE   7,0
9474  052156  050141                  .WORD   DH102
9475  052160  002     000             .BYTE   2,0
9476  052162  047713                  .WORD   DH200
9477  052164  000     000             .BYTE   0,0
9478  052166  050244                  .WORD   DH201
9479  052170  007     000             .BYTE   7,0
9480  052172  050333                  .WORD   DH202
9481  052174  002     000             .BYTE   2,0
9482  052176  051332                  .WORD   DH503
9483  052200  000     000             .BYTE   0,0
9484  052202  047773                  .WORD   DH501
9485  052204  000     000             .BYTE   0,0
9486  052206  050350                  .WORD   DH203
9487  052210  010     000             .BYTE   10,0
9488
9489  052212  000011          DF16:   .WORD   11
9490  052214  000     000             .BYTE   0,0
9491  052216  050052                  .WORD   DH101
9492  052220  007     000             .BYTE   7,0
9493  052222  050141                  .WORD   DH102
9494  052224  002     000             .BYTE   2,0
9495  052226  047713                  .WORD   DH200
9496  052230  000     000             .BYTE   0,0
9497  052232  050244                  .WORD   DH201
9498  052234  007     000             .BYTE   7,0
9499  052236  050333                  .WORD   DH202
9500  052240  002     000             .BYTE   2,0
9501  052242  051271                  .WORD   DH502
9502  052244  000     000             .BYTE   0,0
9503  052246  047773                  .WORD   DH501
9504  052250  000     000             .BYTE   0,0
9505  052252  050350                  .WORD   DH203
9506  052254  010     000             .BYTE   10,0
9507
9508  052256  000005          DF17:   .WORD   5
9509  052260  000     000             .BYTE   0,0
9510  052262  050052                  .WORD   DH101
9511  052264  007     000             .BYTE   7,0
9512  052266  050141                  .WORD   DH102
9513  052270  002     000             .BYTE   2,0
9514  052272  051217                  .WORD   DH711
9515  052274  000     000             .BYTE   0,0
9516  052276  051042                  .WORD   DH702
9517  052300  002     000             .BYTE   2,0
9518
9519  052302  000002          DF20:   .WORD   2
9520  052304  000     000             .BYTE   0,0
9521  052306  050202                  .WORD   DH104
9522  052310  002     000             .BYTE   2,0
9523
9524  052312  000002          DF21:   .WORD   2
9525  052314  000     000             .BYTE   0,0
9526  052316  050650                  .WORD   DH6051
9527  052320  003     000             .BYTE   3,0
9528
```

```
9529  052322  000006              DF22:   .WORD   6
9530  052324     000  000                .BYTE   0,0
9531  052326  047646                      .WORD   DH100
9532  052330     000  000                .BYTE   0,0
9533  052332  050052                      .WORD   DH101
9534  052334     007  000                .BYTE   7,0
9535  052336  050141                      .WORD   DH102
9536  052340     002  000                .BYTE   2,0
9537  052342  050222                      .WORD   DH106
9538  052344     000  000                .BYTE   0,0
9539  052346  050202                      .WORD   DH104
9540  052350     002  000                .BYTE   2,0
9541
9542  052352  000002              DF23:   .WORD   2
9543  052354     000  000                .BYTE   0,0
9544  052356  051407                      .WORD   DH800
9545  052360     001  000                .BYTE   1,0
9546
9547  052362  000001              DF24:   .WORD   1
9548  052364     002  000                .BYTE   2,0
9549
9550  052366  000000              DF25:   .WORD   0
9551  052370     007  000                .BYTE   7,0
9552
9553  052372  000002              DF26:   .WORD   2
9554  052374     002  000                .BYTE   2,0
9555  052376  050350                      .WORD   DH203
9556  052400     010  000                .BYTE   10,0
9557
9558  052402  000005              DF27:   .WORD   5
9559  052404     000  000                .BYTE   0,0
9560  052406  050052                      .WORD   DH101
9561  052410     007  000                .BYTE   7,0
9562  052412  050141                      .WORD   DH102
9563  052414     002  000                .BYTE   2,0
9564  052416  051056                      .WORD   DH703
9565  052420     000  000                .BYTE   0,0
9566  052422  051042                      .WORD   DH702
9567  052424     002  000                .BYTE   2,0
9568
9569  052426  000005              DF30:   .WORD   5
9570  052430     000  000                .BYTE   0,0
9571  052432  050052                      .WORD   DH101
9572  052434     007  000                .BYTE   7,0
9573  052436  050141                      .WORD   DH102
9574  052440     002  000                .BYTE   2,0
9575  052442  051122                      .WORD   DH705
9576  052444     000  000                .BYTE   0,0
9577  052446  051102                      .WORD   DH704
9578  052450     004  000                .BYTE   4,0
9579
9580  052452  000005              DF31:   .WORD   5
9581  052454     000  000                .BYTE   0,0
9582  052456  050052                      .WORD   DH101
9583  052460     007  000                .BYTE   7,0
9584  052462  050141                      .WORD   DH102
```

```
 9585  052464  002  000                    .BYTE   2,0
 9586  052466  051136                      .WORD   DH706
 9587  052470  000  000                    .BYTE   0,0
 9588  052472  051201                      .WORD   DH710
 9589  052474  003  000                    .BYTE   3,0
 9590
 9591  052476  000400              RWBUF:  .BLKW   256.            ;READ/WRITE DATA BUFFER
 9592
 9593  053476  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
 9594  053504  020040  020040  020040
 9595  053512  025052  020052  047516
 9596  053520  042524  025040  025052
 9597  053526  005015     012
 9598  053531     101  046114  042040          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
 9599  053536  044522  042526  020123
 9600  053544  047524  041040  020105
 9601  053552  042524  052123  042105
 9602  053560  046440  051525  020124
 9603  053566  040510  042526  035040
 9604  053574  005015     012
 9605  053577     061  020056  042504          .ASCII  /1. DESIRED PORT SELECTED/<15><12>
 9606  053604  044523  042522  020104
 9607  053612  047520  052122  051440
 9608  053620  046105  041505  042524
 9609  053626  006504     012
 9610  053631     062  020056  051127          .ASCII  /2. WRITE LOCK DISABLED/<15><12>
 9611  053636  052111  020105  047514
 9612  053644  045503  042040  051511
 9613  053652  041101  042514  006504
 9614  053660     012
 9615  053661     104  044522  042526          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
 9616  053666  020123  047516  020124
 9617  053674  047524  041040  020105
 9618  053702  042524  052123  042105
 9619  053710  046440  051525  020124
 9620  053716  040510  042526  041040
 9621  053724  052117  006510     012
 9622  053731     120  051117  051524          .ASCII  /PORTS DE-SELECTED./<15><12>
 9623  053736  042040  026505  042523
 9624  053744  042514  052103  042105
 9625  053752  006456     012
 9626  053755     124  042510  041440          .ASCII  /THE CARTRIDGE MUST BE FORMATTED ON A KNOWN GOOD/<15><12>
 9627  053762  051101  051124  042111
 9628  053770  042507  046440  051525
 9629  053776  020124  042502  043040
 9630  054004  051117  040515  052124
 9631  054012  042105  047440  020116
 9632  054020  020101  047113  053517
 9633  054026  020116  047507  042117
 9634  054034  005015
 9635  054036  042527  046114  040455          .ASCII  /WELL-ALIGNED DRIVE, PRIOR TO TESTING./<15><12>
 9636  054044  044514  047107  042105
 9637  054052  042040  044522  042526
 9638  054060  020054  051120  047511
 9639  054066  020122  047524  052040
 9640  054074  051505  044524  043516
```

```
 9641  054102  006456     012
 9642  054105     124  042510  041440          .ASCII  /THE COMPLETE RUN TIME IS 8-10 MIN. PER DRIVE./<15><12>
 9643  054112  046517  046120  052105
 9644  054120  020105  052522  020116
 9645  054126  044524  042515  044440
 9646  054134  020123  026470  030061
 9647  054142  046440  047111  020056
 9648  054150  042520  020122  051104
 9649  054156  053111  027105  005015
 9650  054164  047524  040440  047502          .ASCIZ  /TO ABORT TESTING, TYPE CONTROL-C (↑C)./<15><12><12>
 9651  054172  052122  052040  051505
 9652  054200  044524  043516  020054
 9653  054206  054524  042520  041440
 9654  054214  047117  051124  046117
 9655  054222  041455  024040  041536
 9656  054230  027051  005015  000012
 9657
 9658
 9659
 9660
 9661
 9662
 9663          000001                  .END
```

```
ABASE = 177440          1464#  1560   1601   3522
ABORT = 000200          3177#  3756   4830   6478   6483   6566   6591   6688   6698   6703   6708   6713   6724
                        6730   6745   6780
ACDW1 = 000000          1560
ACDW2 = 000000          1560
ACLO  = 000010          2523#  6528
ACPUOP= 000000          1560   1575
ADDW0 = 000000          1560
ADDW1 = 000000          1560
ADDW10= 000000          1560
ADDW11= 000000          1560
ADDW12= 000000          1560
ADDW13= 000000          1560
ADDW14= 000000          1560
ADDW15= 000000          1560
ADDW2 = 000000          1560
ADDW3 = 000000          1560
ADDW4 = 000000          1560
ADDW5 = 000000          1560
ADDW6 = 000000          1560
ADDW7 = 000000          1560
ADDW8 = 000000          1560
ADDW9 = 000000          1560
ADEVCT= 000000          1560   1566
ADEVM = 000377          1465#  1560   1602
ADSCOR  017272          4013   4083   4145   4432#
AENV  = 000000          1560   1571
AENVM = 000000          1560   1572
AFATAL= 000000          1560   1563
ALLSYS  012546          3516   3539#  3552   3553   3596   3626   3627   3644   4945   5017   5640
ALLSY1  012604          3549#  3633
ALLSY2  012766          3589#  3599
ALLSY3  012764          3588#  3736
ALLTRM  032472          3757   4831   6567   6711   6718#
ALNPAK  011136          3343#  4886
AMADR1= 000100          1560   1588
AMADR2= 000000          1560   1592
AMADR3= 000000          1560   1595
AMADR4= 000000          1560   1598
AMAMS1= 000000          1560   1582
AMAMS2= 000000          1560   1590
AMAMS3= 000000          1560   1593
AMAMS4= 000000          1560   1596
AMSGAD= 000000          1560   1568
AMSGLG= 000000          1560   1569
AMSGTY= 000000          1560   1562
AMTYP1= 000000          1560   1583
AMTYP2= 000000          1560   1591
AMTYP3= 000000          1560   1594
AMTYP4= 000000          1560   1597
ANOTHR  007744          3223#  3624
ANYDER= 100000          3183#  4878   6201   6768
APASS = 000000          1560   1565
APRIOR= 000000          1560
APTCSU= 000040          8108   8753#
APTENV= 000001          8101   8547   8709   8751#
```

```
APTSIZ= 000200  3482  8750#
APTSPO= 000100  8103  8711  8752#
ASKSYS  013212  3619  3630  3635  3641#  3645  3646  3648  3653  3664  4949  5021  5654
ASWREG= 000000  1560  1573
AS2SP2  044546  6240  8844#
ATESTN= 000000  1560  1564
AUNIT = 000000  1560  1567
AUSWR = 000000  1560  1574
AVECT1= 120210  1463#  1560  1599  3521
AVECT2= 000000  1560  1600
AVSCOR  017452  4021  4090  4154  4474#
A.ABNL  003046  2804#  3760*  3763*  4766*  4834*  4839*  5645*  5827*  5844*  5853*  6037*  6052*  6454*
                6719*  6722*  6736*  6739*  6742*  7669
A.CONT  003050  2805#  7679
A.NORM  003044  2803#  4767*  6455*  6723*  7673
BADDRV  011053  3332#  4278*  4279*  4283  4837*  4842*  4843*  4844  4853*  4854*  4855  4861*  4862*
                4863  4883*  4884*  4885
BADSEC= 001000  3179#  5569  5754  6117  6125  6661  6773
BAD632  011606  3402#  4880
BAI   = 000020  2482#  7838
BA16  = 000400  2458#
BA17  = 001000  2459#
BDSCHD  026140  5827  5844  5858#
BDSECT  011557  3396#  5876
BDSRCK  027176  5567  5752  6107#  6660
BGNRTY  032714  6378  6733  6744  6766#
BIT0  = 000001  1395#  2455  2472  2498  2520  2668  2812  2905  3716  3745  4819  5489  5538
                5578  5765  6337  7953
BIT00 = 000001  1385#  1395
BIT01 = 000002  1384#  1394
BIT02 = 000004  1383#  1393
BIT03 = 000010  1382#  1392
BIT04 = 000020  1381#  1391
BIT05 = 000040  1380#  1390
BIT06 = 000100  1379#  1389
BIT07 = 000200  1378#  1388
BIT08 = 000400  1377#  1387  5380
BIT09 = 001000  1376#  1386  8557  8683
BIT1  = 000002  1394#  2473  2500  2669  2813  2906  3171  6341
BIT10 = 002000  1375#  2460  2489  2511  2546  2560  2573  2588  2602  2681  2822  3180  6370
                8535
BIT11 = 004000  1374#  2461  2490  2512  2531  2547  2561  2574  2589  2603  2682  3181
BIT12 = 010000  1373#  2463  2491  2513  2548  2562  2575  2576  2590  2604  2683  3182  6252
                6280
BIT13 = 020000  1372#  2464  2492  2514  2532  2549  2563  2577  2591  2605  2684  8542
BIT14 = 040000  1371#  2465  2493  2515  2533  2550  2564  2578  2592  2606  2685  2823  8667
BIT15 = 100000  1370#  2466  2467  2494  2516  2534  2551  2611  2686  2824  3183  6373
BIT2  = 000004  1393#  2474  2501  2502  2522  2670  2814  2907  3172  6345
BIT3  = 000010  1392#  2480  2481  2503  2523  2671  2815  2908  3173  6352
BIT4  = 000020  1391#  2475  2482  2504  2524  2525  2540  2582  2596  2672  2816  2909  3174
                6356
BIT5  = 000040  1390#  2483  2484  2505  2526  2541  2555  2568  2583  2597  2673  2817  3175
                4638  6360  7750
BIT6  = 000100  1389#  2456  2485  2506  2527  2542  2556  2569  2584  2598  2674  2819  3176
                4643  4676  5605  5792
BIT7  = 000200  1388#  2457  2486  2507  2528  2529  2543  2557  2570  2585  2599  2676  3177
```

```
                    3519   3713   3726   4982
BIT8  = 000400      1387#  2458   2487   2508   2509   2530   2544   2558   2571   2586   2600   2677   2820
                    3178   6364
BIT9  = 001000      1386#  2459   2488   2510   2545   2559   2572   2587   2601   2678   2821   3179   4971
                    6367   8647
BKSLSH  011707      3414#  5271   5279
BLDEXH  033600      6607   6637   6920#
BPTVEC= 000014      1402#
BSE   = 000200      2507#  5858   6652   7103   7269   7505
BSERR = 000002      3171#  6040   6654   6668   6766
BSFACT  004152      2915#  5826   5874   6108
BSSOFT  003152      2914#  5838   5907   6113   6115
BUFFO   005202      2920#  3559   3563   3565   3570   3581   3583   3585   3631   3634   3636   3649   3651
                    3654   3656   3697   4246*  4247*  4249*  4251   4254   4259   4264*  4281*  4286   4290*
                    4719   4723   4938   4950   5010   5022   5274   5288*  5293*  5294   5298*
BUFPRT  005412      2923#
B.BA16= 000001      2472#  7832
B.BA17= 000002      2473#  7832
B.CDT = 000004      2474#  7486   7832
B.CFMT= 000020      2475#  4973   6876   7486   7832   7916
CART    011510      3388#  5357
CCLR  = 100000      2467#  7439
CDT   = 002000      2460#  7245   7795   7852   7903   7926
CERR  = 100000      2466#  6616   6646   7141   7200   7259   7441   7492   7807   7863   7951
CFMT  = 010000      2463#  6899   6932   7245   7795   7852   7903   7926
CKMEM1  025250      5729#
CKMEM2  025262      5733#
CKMO    025270      5732   5735#
CKSCOR  022602      5159#  6043
CLEAR = 000105      2426#
CLR   = 000040      2483#
CLRPRM  020710      4771   4788#
CMDTO = 000100      2674#  6501   6831   7002
CMPBUF  024202      5525#
CMSTRT  012006      3438   3443#  6757
CNTRDY  011307      3363#  5659
CNTRLC  011652      3408#  4642   5242
CNTRLG  011676      3412#  4656   5260
CNTRLR  011664      3410#  4652
CNTRLU  011671      3411#  5254
CNTRLZ  011657      3409#  4648   5248
COE   = 001000      2510#  6573   7103   7269   7505
COMAVG  007266      3143#  4290   4532*  4544*  4545*  4572
COMMA   011711      3415#  3609
COMSTR  005166      2918#  6172   6181   6785
CONCLR= 000176      2444#  7934
CONERR  030134      2805   6333#
CR    = 000015      1310#  8147   8157
CRLF  = 000200      1311#  8118   8157
CRTSER  023440      4981   5346#
CR2LF   011703      3413#  6239   6251   6275
CTLOUT  024722      4235   5632#  6153
CTO   = 004000      2461#  6536   7501
CYLNDR  005424      2928#
C.INIT  037356      6158   7712#
C.RTRN  041230      7816   7844   7872   7879   7882   7889   7893   7910   7932   7937   7947   7956   7969
```

```
                    7973    7978#
C.SPEC  040250      7752    7848#
DCEFLG  003134      2892#   4622*   6035*
DCK     = 100000    2516#   4618    5858    6670    7103    7269    7505
DCKERR= 000020      3174#   4621    5162    5975    6040    6672    6766
DCKHDL  020204      4613#   6037    6719    6739
DCLO    = 000020    2525#
DDISP   = 177570    1317#   1494    3470
DDT     = 000400    2530#
DERCNT  003120      2880#   6600    6630    6679    6690
DESL    = 000010    2480#
DFLTDR  013540      3529    3711#
DFSTRT  011746      1421    3433#   4944    5016
DF01    051542      1625    1631    1637    1643    1685    1793    2063    2099    9328#
DF02    051572      1649    1667    1673    1679    1691    1697    1703    1709    1715    1721    1727    1733    1739
                    1745    1751    1757    1811    1817    1823    1931    1937    1943    1949    1955    1961    1967
                    1973    1979    1985    2027    2045    2069    9342#
DF03    051626      1655    1661    2033    2039    9357#
DF04    051666      1787    9374#
DF05    051702      1763    1769    9381#
DF07    051736      1775    9396#
DF10    051772      1781    9411#
DF11    052016      2081    9422#
DF12    052036      1829    1835    1841    1847    1853    1859    1865    1871    9431#
DF13    052066      1877    1889    9444#
DF14    052116      1883    9457#
DF15    052146      1895    9470#
DF16    052212      1901    9489#
DF17    052256      1919    9508#
DF20    052302      2087    9519#
DF21    052312      1997    9524#
DF22    052322      2093    9529#
DF23    052352      2003    2009    2015    9542#
DF24    052362      2021    9547#
DF25    052366      1925    5527*   5557*   5739*   5757*   9550#
DF26    052372      2075    9553#
DF27    052402      2051    9558#
DF30    052426      2105    9569#
DF31    052452      2111    9580#
DH100   047646      1623    1629    1635    1641    1647    1653    1659    1665    1671    1677    1683    1689    1695
                    1701    1707    1713    1719    1725    1731    1737    1743    1749    1755    1761    1767    1773
                    1779    1785    1791    1809    1815    1821    1827    1833    1839    1845    1851    1857    1863
                    1869    1875    1881    1887    1893    1899    1917    1929    1935    1941    1947    1953    1959
                    1965    1971    1977    1983    2025    2031    2037    2043    2049    2061    2067    2097    2103
                    2109    9145#   9531
DH101   050052      6256    9170#   9331    9344    9359    9376    9383    9398    9413    9433    9446    9459    9472
                    9491    9510    9533    9560    9571    9582
DH102   050141      6267    9180#   9333    9346    9361    9378    9385    9400    9415    9435    9448    9461    9474
                    9493    9512    9535    9562    9573    9584
DH103   050157      2085    9183#
DH104   050202      9187#   9521    9539
DH105   047670      6254    9148#
DH106   050222      9190#   9537
DH200   047713      9152#   9335    9348    9363    9437    9476    9495
DH201   050244      9193#   9337    9350    9365    9439    9478    9497
DH202   050333      9203#   9352    9369    9480    9499
```

```
DH203   050350          9206#   9354    9371    9486    9505    9555
DH204   051233          9287#   9441
DH500   047735          9156#   9339
DH501   047773          9162#   9367    9484    9503
DH502   051271          9293#   9454    9467    9501
DH503   051332          9299#   9482
DH601   050444          9216#   9387    9450    9463
DH602   050466          9219#   9391
DH604   050533          9226#   9402    9417    9424
DH6041  050566          5877    9231#   9404    9419    9426
DH6042  050615          2019    9235#
DH605   050633          1995    9238#   9406
DH6051  050650          9241#   9408    9526
DH606   050705          9246#   9389    9393    9452    9465
DH607   050733          9250#
DH701   050753          5526*   5556*   5738*   5756*   9253#   9428
DH702   051042          2073    9263#   9516    9566
DH703   051056          9265#   9564
DH704   051102          9269#   9577
DH705   051122          9272#   9575
DH706   051136          9274#   9586
DH707   051151          9276#
DH710   051201          9281#   9588
DH711   051217          9284#   9514
DH800   051407          9307#   9544
DI    = 040000          2465#   7289    7961
DIF100  003137          2895#   3979*   4015*   4056*   4128*   4151*   4369    4374*
DISPAK  010154          3247#   4997*   4998*   5001
DISPLA  001142          1494#   3470*   3478*   8534*   8694*
DISPRE  000174          1418#   3478
DLT   = 100000          2494#   6593    7097    7227    7263    7503
DLTFLG  003136          2894#
DMCART  022054          3900    4297    4995#
DMD   = 000040          2541#
DONE    003122          2882#   6154*   6161    6200*   6334*   6448*   6800*   6803*
DRA   = 000001          2520#   7092    7875
DRDY  = 000200          2529#
DRIDON  010412          3275#   5003
DRIRDY  010373          3272#   4931
DRIV    011501          3386#   5315
DRIVE   005420          2926#   3741*   3744    4593    4774    4775    4818    4842    4853    4861    4883    4954*   4955*
                        9312
DRNAFG  003131          2889#   6155*   6347
DRNAVL= 010000          3182#
DROPDR  011411          3376#
DROT  = 000040          2526#   6583
DROVFL  010015          3230#   3595
DRPAR = 000010          2503#   6523
DRPDRV= 002000          2681#   7737
DRVBUF  011000          3321#   4242*   4244*   4245
DRVCAL  027302          3762    3831    3834    3983    4074    4077    4372    4375    4405    4411    4414    4773    4836
                        4868    4870    4877    4904    4962    5314    5356    5648    5689    5829    5846    6039    6151#
                        6737    6784    6794    6830    6867    6869
DRVDSC= 000040          2673#   6700    7177    7343    7405
DRVERS  003127          2887#   3490*   6224*
DRVHRD= 000020          2672#   6695    7184    7379    7510
```

```
DRVLST  005502          2946#  3542   3547   3593   3657   3661   3728   3913   4159   4179   4232   4311   4529
                        4924   5047
DRVMSK= 000007          2479#  7086   7127   7278   7723
DRVNO   011475          3385#
DRVPDT= 000004          2670#  7285   7353
DRVPOS= 000002          2669#  7285   7312   7774
DRVPTR  005544          2950#  3666*  3731*  3901*  3902   3913*  4226   4238   4277   4298*  4299   4311*  4921
                        4954   4996
DRVSEQ  011035          3329#
DRVSER  023320          4980   5311#
DRVSZD= 010000          2683#  6514   7115   7877
DRVTST  013500          3622   3696   3701#
DRVUSE= 000001          2668#  7737
DRY     = 000200        2528#
DR.CLR= 000005          2451#  7165   7328   7370   7394   7508
DR.SEL= 000001          2450#  7235   7314   7629   7636   7643   7650   7854
DROXDP  011360          3371#
DSC     = 040000        2533#
DSWR    = 177570        1316#  1493   3469
DTBAII= 100000          2686#  3992   4137   5983   7737   7836
DTE     = 010000        2513#  5858   6588   7505
DTYE    = 000040        2505#  6551   7505
DT100   051432          1624   1630   1636   1642   1648   1654   1660   1666   1672   1678   1684   1690   1696
                        1702   1708   1714   1720   1726   1732   1738   1744   1750   1756   1762   1768   1774
                        1780   1786   1792   1810   1816   1822   1828   1834   1840   1846   1852   1858   1864
                        1870   1876   1882   1888   1894   1900   1918   1930   1936   1942   1948   1954   1960
                        1966   1972   1978   1984   2026   2032   2038   2044   2050   2062   2068   2092   2098
                        2104   2110   9312#
DT102   051450          9315#
DT201   051454          9316#
DT202   051472          2074   9319#
DT203   051476          9320#
DT601   051516          1996   2002   2008   2014   2020   2080   9323#
DT602   051524          1924   9324#
DZR6Q   007430          3185#  3494
ECCNC   = 000040        3175#  6675
ECCW    = 020000        2549#
ECH     = 000100        2506#  6673   7103   7269   7505
ECOBAD  020442          4690#
EMTVEC= 000030          1405#  3454*  3455*
EM1     044553          1622   8845#
EM10    044766          1664   8872#
EM100   047077          2012   9079#
EM101   047122          2018   9083#
EM102   047200          2024   9091#
EM103   047252          2030   9098#
EM104   047300          2036   9102#
EM105   047322          2042   9105#
EM106   047355          2048   9110#
EM107   047410          2054   9115#
EM11    044775          1670   8874#
EM110   047455          2060   9122#
EM111   047510          2066   9127#
EM112   047531          2072   9130#
EM113   047564          2096   9135#
EM114   047612          2102   9139#
```

```
EM115   047627          2108    9142#
EM12    045010          1676    8876#
EM13    045031          1682    8879#
EM14    045053          1688    8883#
EM15    045101          1694    8887#
EM16    045122          1700    8890#
EM17    045137          1706    8893#
EM2     044577          1628    8849#
EM20    045160          1712    8896#
EM21    045175          1718    8899#
EM22    045215          1724    8902#
EM23    045237          1730    8906#
EM24    045270          1736    8911#
EM25    045310          1742    8914#
EM26    045333          1748    8918#
EM27    045345          1754    8920#
EM3     044623          1634    8853#
EM30    045370          1760    1880    8924#
EM31    045415          1766    1886    8928#
EM32    045436          1772    8931#
EM33    045457          1778    8934#
EM34    045501          1784    2084    8938#
EM35    045521          1790    8941#
EM36    045555          1796    8946#
EM37    045610          1802    8951#
EM4     044646          1640    8857#
EM40    045653          1808    8957#
EM41    045717          1814    8964#
EM42    045755          1820    8970#
EM43    046005          1826    8975#
EM5     044667          1646    8860#
EM52    046042          1868    8980#
EM53    046070          1874    8984#
EM56    046115          1892    1898    8988#
EM6     044706          1652    8863#
EM60    046134          1904    8991#
EM61    046170          1910    8996#
EM62    046236          1916    9003#
EM63    046262          1832    1928    9007#
EM64    046327          1838    1934    9014#
EM65    046372          1844    1940    9020#
EM66    046420          1850    1946    9024#
EM67    046453          1856    1952    9029#
EM7     044732          1658    8867#
EM70    046512          1862    1958    9035#
EM71    046561          1964    9042#
EM72    046621          1970    9048#
EM73    046670          1976    9055#
EM74    046724          1982    9060#
EM75    046744          1988    9063#
EM76    046774          1994    9067#
EM77    047056          2000    2006    9076#
ENDTST  010740          3315#   4310
EQUALS  011734          3424#
ERRCNT  003125          2885#   3491#   4901#   6038#   6203#   6205    6208    6210#   6335#   6449#   6620#   6665#   6755#
                        6775    6778    6795#
```

```
ERRCOM  005416        2925#  6386*  6387*  9312
ERRFRE  027426        2803   4767   6200#  6723
ERRHDL  030640        2804   3763   4588   4620   4766   4839   5645   5853   5860   6052   6447#  6722   6742
ERRLMT  003126        2886#  6775
ERRVEC= 000004        1398#  3467   3468*  3479*  3487*  3488*  8672   8673*  8675*  8678*
ER2ENT  030660        6451#  6513   6626   6651   6747
E.CCLR= 000001        2812#  7444
E.CERR= 001000        2821#  7518
E.CLAT= 000020        2816#  7170   7334   7375   7399
E.CMTO= 040000        2823#
E.CONT  003052        2806#  6337   6341   6345   6352   6356   6360   6364   6367   6370   6373   6377*  7069*
                      7107*  7170*  7180*  7251*  7273*  7296*  7302*  7334*  7375*  7399*  7444*  7496*  7518*
                      7657*  7891*  7945*  7953*  7966*
E.DLT = 000400        2820#  7251
E.DPAR= 002000        2822#  7180   7657
E.ILLD= 000100        2819#  7945
E.MDS = 100000        2824#  7069   7496   7891
E.NOAT= 000002        2813#  7296
E.SCLR= 000040        2817#  7966
E.UATT= 000004        2814#  7302
E.UDAT= 000010        2815#  7107   7273
E.UNLD= 020000        2684#  7188   7308   7383   7393
FACTBS  011532        3392#  5875
FINADR  023700        5432#  5562   5747
FINCYL  005460        2938#  5462*  5564   5582   5588   5749   5769   5775
FINMEM  026336        5919#  5970
FINSEC  005463        2940#  5450*  5566   5586   5590   5751   5773   5777
FINTRK  005462        2939#  5460*  5565   5584   5589   5750   5771   5776
FMTE  = 000020        2504#  6555   7505
FORMAT  003124        2884#  3994   4139   4776   4969*  4973*  5348   5442   5821   5841   5946   6877
GETPRM  020454        3670   3675   3684   4705#  4745
GMS   = ****** U      1417   8830   8831   8832   8833   8834   8837   8838   8839   8840
GO    = 000001        2455#
GTPKAD  033444        5165   6681   6692   6887#
GTPRMS  026674        6012#  6049
GTSWRG  020564        3512   4741#  5261   5665
GTO     026706        6011   6015#
HCRC  = 000400        2508#
HDR.AD  003070        2846#  7221   7225*  7915*
HDR.CT  003072        2847#  7229*  7918*  7921*
HIGOCT  005410        2922#
HLPOVL  003142        2899#  3493*
HLTPRG  032676        5660   5837   6753#  6781
HLTRQD  011337        3368#  5658
HT    = 000011        1308#  8116   8157
HVRC  = 000400        2509#  4614   5858   6627   7103   7269   7505
HVRCER= 000004        3172#  4616   5162   6040   6629   6766   6892
H.HEAD  003075        2850#
IDAE  = 002000        2511#  6578
IE    = 000100        2456#  3743   4817   4847   4928   5000   6612   6642   7449   7513   7801   7842   7908
                      7975
ILC   = 000001        2498#  6539   7505
ILF   = 000004        2501#  6547
INCRSC  026422        5931   5945#  5978
INITSS  020616        3742   3758   3792   3815   3855   3897   3942   3976   4048   4125   4222   4766#  4816
                      4832   4927   4952   4960   4974   4999   5024   5312   5346   5815
```

```
INTBLD  022224          3857    3869    4011    4134    5037#
INTCHR  005432          2931#   4639*   4658    4675*   4690    4985*   5634    5638    5643*   5652    5656    5663
INTMSK  003101          2855#   7144    7168    7189    7214    7300    7332    7373    7384    7397    7668*   7672*   7676*
                        7725*   7811
INTR   = 000300         2446#
IOTVEC= 000020          1403#   3452*   3453*
IR     = 000100         2485#
I.AERR  035740          7320    7326    7344    7353#
I.ATTN  035270          7080    7124    7257#
I.CCLR  036324          7158    7208    7439#   7511    7871    7936
I.CSTS  036756          7151    7586#
I.CST1  037000          3753    4827    7155    7207    7590#   7860
I.DRV   003102          2859#   7725    7726
I.ERR   034512          7152    7156#
I.ERRA  034472          7151#   7219
I.ERRC  034500          7089    7142    7154#   7287
I.HDAL  034736          7135    7198#
I.INTR  034056          3502    7059#
I.ISRL  003074          2849#   7073    7076*   7079*   7450*   7747*   7901*   7974*
I.ISSU  036406          7166    7236    7315    7329    7371    7395    7481#   7630    7637    7644    7651
I.IOO   034336          7077    7094    7121#
I.RTRN  036306          7071    7109    7119    7147    7159    7164    7167    7173    7182    7191    7194    7209    7211
                        7237    7240    7249    7253    7275    7291    7298    7304    7316    7330    7337    7349    7369
                        7372    7377    7386    7389    7396    7401    7410#
I.STAT  037062          7163    7368    7627#   7881
I.STOR  036630          7491    7541#
I.UNLD  036206          7310    7393#
KBDHDL  020262          3499    4631#
LASTWC  005464          2941#   5433    5561*   5746*
LDMEM1  025200          5701#
LDMEM2  025212          5705#
LDMO    025220          5704    5707#
LEV2ER= 000400          3178#   6451    6506    6603    6609    6633    6639    6743    6801
LF     = 000012         1309#   8151    8157
LOOBUF  024056          5480#
LOOSEC  023640          3819    4061    5404#
LOGDRV  003144          2901#   3821    3986    4009    4058    4926*   5121    5127
LOWOCT  005406          2921#
LSTCYL= 000632          3169#   4872    5352
LSTTRK= 000002          3170#   4873    5353
MA      005470          2943#
MCLK   = 000400         2544#
MDFLAG  003116          2878#   3433*   3436*   3440*   3515    3620    3914    4312    4942    4946    4976    5014    5018
                        5636
MDS    = 001000         2488#   3749    4823    7067    7494    7869
MEMABT  003141          2897#
MEMVEC= 000114          2912#
MERD   = 001000         2545#
MESMSK= 000017          2538#
MEMD   = 002000         2546#
MIDXFR  026466          5969#   6044
MIND   = 000200         2543#
MNTPAK  010110          3241#   4922*   4923*   4929
MSP    = 000100         2542#
MTCART  021472          3794    3945    4920#
M.CADD= 017760          2614#
```

```
M.CDIF= 017760          2613#
M.DPID  042032          4500    4565    8200#
M.DPIM  041772          8171#
M.DPO1  042002          8174#   8183
M.DPO2  042022          8178    8182#
M.DP40  042070          8211#   8232
M.DP41  042124          8216    8223#
M.DP42  042134          8222    8227#
M.DP44  042166          8234    8239#
M.DP50  042200          8209    8243#
M.DRV = 000007          2610#
M.HEAD= 007000          2617#
M.ID  = 000003          2612#
M.PAR = 100000          2611#
M.SECT= 000760          2616#
M.SER = 077770          2615#
NCSCR0  006750          3095#   4129    4131    4157    4174    4452    4487    4534
NCSCR1  007010          3098#
NCSCR2  007050          3101#
NED   = 010000          2491#   4586    6485    6495    7082    7160    7503    7873
NEDHDL  020126          3760    4586#   4834
NEM   = 004000          2490#   6480    7097    7263    7503
NEWMSG  011024          3327#   4709
NEWON   005466          2942#   3761*   3764    4598*   4835*   4840
NGOFST  005442          2935#
NOCHK = 000400          2677#   7737^   7799    7840    7867    7885    7906    7929    7938    7972
NOORLS  010422          3277#   3663
NOORTS  011256          3358#
NOOSC = 004000          2682#   6705    7325
NOFALS  011577          3399#   5884
NORTRY  003140          2896#   4770*   6771
NOSCR0  006450          3073#   4004    4006    4230    4448    4482    4533    4550
NOSCR1  006510          3076#
NOSCR2  006550          3079#
NOTMSG  053476          3497    9593#
NOTYPE  003143          2900#   3495    3498*
NSCOR0  007270          3148#   4065*   4363    4365    4387    4393    4435    4455    5189
NSCOR1  007310          3151#   4066*   4437
NSCOR2  007330          3154#   4067*   4439
NSSCR0  007250          3118#   4050*   4158    4436*   4477
NSSCR1  007252          3121#   4051*   4438*
NSSCR2  007254          3124#   4052*   4440*
NTREDY  011103          3337#   4856
NXDRIV  011064          3334#   4845
NXF   = 000004          2502#
OAVERG  010702          3309#   4282
OCTBIN  041252          4724    8003#
OFFSET= 000115          2430#   4404    4413    7782
OFINUS  005426          2929#
OFSCNT  003147          2904#   4379*   4398    4406*   5200
OFSDIR  003146          2903#   4376*   4388    4395    4420*   4491*   4497    4506*   5190
OFST  = 000004          2522#
ONSUBS  007724          3220#   3615*   3616
OPCOMP  003121          2881#
OPCONT  003130          2888#
OPI   = 020000          2514#   5858    6597    7103    7269    7505
```

```
OPIERR= 000010          3173#  6599   6766
OR    = 000200          2486#
OREQSW= 000020          2909#
OVRAVG  007264          3140#  4281   4531*  4542*  4543*  4559
O.WAIT  003054          2826#  7003   7007*  7117*  7121   7137*  7203*  7216*  7231*  7775*  7835*  7897*  7914*
PACK  = 000103          2425#  4867
PARM0   002630          2715#  2870   4769   4902   6005   6014   6718   6734   6738*  6792   6844   6878
PARM1   002714          2747#  4775*  6453   6734*  6738   6828   6862
PASSNO  003145          2902#  3437*  3441*  3701   3703   3944*  4978   5661*
PAT   = 000020          2540#
PATRN   005436          2933#  5482   5531   5607
PATOO   005556          2953#  5484   5499   5533   5548
PBLKT   003112          2870#  7000   7087   7113   7281   7306   7724*
PBSVAL= 001000          2678#  7662
PCA   = 004000          2547#
PCD   = 010000          2548#
PCSCR0  007110          3106#  4177   4194   4453   4488
PCSCR1  007150          3109#
PCSCR2  00721C          3112#
PGE   = 002000          2489#  6543   7503
PIP   = 020000          2532#
PIRQ  = 177772          1315#
PIRQVE= 000240          1409#
PLOFST  005440          2934#
PMA     005474          2944#  5501   5550   5599   5601   5708   5736   5786   5788   5988*  5989*  5990*  5991
                        5992   6009*  6010*
POSCR0  006610          3084#  4449   4483
POSCR1  006650          3087#
POSCR2  006710          3090#
PREPKB  020422          3793   3816   3856   3898   3943   3977   4049   4126   4223   4674#  5666
PRHBUF  011722          3422#
PRMPSP  011743          3426#
PROMPT  011740          3425#  4256
PRVCMD  005152          2916#  3506   6171   6420
PR0   = 000000          1332#  3505   4905   6793   6866
PR1   = 000040          1333#
PR2   = 000100          1334#
PR3   = 000140          1335#
PR4   = 000200          1336#  3500
PR5   = 000240          1337#  2802
PR6   = 000300          1338#
PR7   = 000340          1339#  3443   8627   8628
PS    = 177776          1312#  1313   3443*  3505*  4905*  6989   6995*  7009*  7718   7719*  7978*
PSCOR0  007350          3159#  4068*  4397   4441   4456   5192
PSCOR1  007370          3162#  4069*  4443
PSCOR2  007410          3165#  4070*  4445
PSSCR0  007256          3129#  4053*  4178   4442*  4478
PSSCR1  007260          3132#  4054*  4444*
PSSCR2  007262          3135#  4055*  4446*
PSW   = 177776          1313#  6793*  6866*
PWRMSG  043674          8630   8634#
PWRVEC= 000024          1404#  3458*  3459*  8618*  8626*  8627*
P.ASOF= 000032          2699#  6407   7331*  7332   7365*  7596*  7784*  7785
P.A00 = 000040          2702#  4851   4859   6410   6834   7322*  7323   7339*  7340   7653*  7861*
P.A01 = 000044          2704#  6412   7632*
P.A10 = 000050          2706#  6414   7639*
```

```
P.A11 = 000054   2708#  5317   6416   7646#
P.BAHI= 000007   2661#  5994#  6396
P.BALO= 000010   2662#  3827#  3991#  4060#  4136#  4874#  5354#  5685#  5826#  5838#  5991#  6399  7818
                 7915
P.BAR = 000024   2696#  6406   7362#  7592#
P.B00 = 000042   2703#  6411   7317#  7318   7654#  7862#
P.B01 = 000046   2705#  6413   7633#
P.B10 = 000052   2707#  6415   6856   7640#
P.B11 = 000056   2709#  6417   7238#  7647#
P.CMND= 000001   2655#  3751#  3759#  3830#  3833#  3981#  3985#  4073#  4076#  4368#  4404#  4407#  4410#
                 4413#  4772#  4825#  4833#  4867#  4869#  4871#  4903#  4961#  5313#  5351#  5647#  5688#
                 5817#  6387   6735#  6783#  6829#  6865#  6868#  7133   7750   7754   7766   7776   7782
                 7788   7791   7797   7822   7829   7848   7895   7912   7934   7943
P.CS1 = 000016   2693#  3752#  4791   4826#  6400   6518   6536   6899   6932   7154#  7206#  7244#  7245#
                 7247#  7248   7357#  7359#  7586#  7589#  7740   7794#  7795#  7797#  7801#  7802   7826#
                 7829#  7831#  7832#  7842#  7843   7851#  7852#  7854#  7855   7857#  7858   7863   7902#
                 7903#  7905#  7908#  7909   7925#  7926#  7928#  7931   7950#  7951   7960#  7961
P.CS1H= 000007   2660#  4776#  4876#  5347#  5816#  6876#  6877#  7244   7485   7794   7831   7851   7902
                 7916   7925
P.CS2 = 000020   2694#  4586   6401   6475   6480   6485   6487   6493   6495   6543   6593   6686   7160
                 7360#  7590#  7748#  7773   7838#  7839   7850   7869   7873   7899#  7900   7924
P.CYLN= 000002   2656#  3824#  3859#  3862#  3870#  3982#  3987#  3999#  4010#  4014#  4072#  4075#  4143#
                 4146#  4371#  4373#  4872#  5138#  5352#  5461   5818#  5920   5979#  6389   6864#  7242
                 7778   7820   7922
P.DCYL= 000030   2698#  5924   5971   6402   6655   6887   6921   7364#  7598#
P.DRVN= 000000   2654#  4774#  7130   7307   7483   7722
P.DS  = 000036   2701#  6408   6528   6532   6583   7367#  7594#  7875
P.DTS = 000026   2697#  5926   5928   5972   5973   6403   6656   6658   6855   6890   6891   6922   6928
                 7363#  7593#
P.EPAT= 000062   2711#  6683   7156#  7742
P.EPOS= 000060   2710#  6682   7157#
P.ER  = 000034   2700#  4614   4618   5858   6409   6474   7366#  7595#
P.OFST= 000006   2659#  4403#  4412#  7784
P.PRST= 000014   2664#  3992#  4137#  5983   6501   6514   6695   6700   6705   6710   6831   7002#  7115#
                 7149#  7177#  7184   7188#  7285#  7308   7312#  7325#  7343#  7353#  7379   7383#  7393#
                 7405#  7510#  7662#  7737#  7774#  7798#  7799   7824   7830#  7836   7840   7867   7877#
                 7885   7906   7929   7938   7972
P.SECT= 000004   2657#  3826#  4063#  4084#  4085   5131#  5350#  5438   5823#  5832#  5833   5839#  5849#
                 5850   5922   5981#  6392   6863#  7243   7779   7821   7923
P.TRCK= 000005   2658#  3836#  3837   4064#  4080#  4081   4382   4873#  5132#  5134#  5135   5353#  5452
                 5819#  5921   5980#  6390
P.WC  = 000012   2663#  3828#  3993#  3996#  4062#  4138#  4141#  4875#  5355#  5686#  5825#  5982#  6007
                 6045   6394   7819
P.WCR = 000022   2695#  6404   7361#  7591#
P1SET   011762   3434   3437#
P1STRT  011754   1423   3436#
P2STRT  011772   1425   3440#
QNEWSW= 000000   2414
Q.INIT= 040000   2685#
RAVERG  010721   3312#  4291
RCOASW= 000010   2908#
RCLREQ= 004000   3181#  6521   6526   6571   6576   6581   6586   6623   6732
RDALHD= 000164   2443#  7133   7912
RDCHR = 104406   5237   8837#
RDCHRS  023006   3551   3625   3643   4710   4932   5004   5231#
RDDATA= 000121   2432#  4407   4871   5351   5817
```

```
RDGATE= 100000  2551#
RDHDO   033270  6608    6638    6853#
RDHEAD= 000125  2434#   4961    6868    7247    7928
RDSTAT= 000141  2442#   3759    4833    5313    6829    7848
RDY   = 000200  2457#   4964    4967    6728    6871    6874    7805    7858
RECAL = 000113  2429#   4869    5647    6735    7791
RECODE  005414  2924#   3756*   4613*   4616*   4621*   4830*   4878    4900*   5162    5569    5754    5975    6036*
                6040    6117*   6125*   6201    6211*   6450*   6451    6478*   6483*   6504    6506    6521*   6526*
                6566*   6571*   6576*   6581*   6586*   6591*   6599*   6603    6609    6623*   6629*   6633    6639
                6654*   6661    6668*   6672*   6675*   6688*   6689*   6698*   6703*   6708*   6713*   6724    6730*
                6732    6743    6745*   6766    6768*   6773    6780*   6801*   6833*   6892
REDBSF  025676  4975    5814#
REDOFS  016726  4012    4079    4144    4359#
REISSU  003132  2890#   6349*
RELEAS= 000140  2441#   7895
REPLPK  011167  3348#
REPSUP  030360  3754    4828    5530    5740    5835    6384#   6456
RESLTS  010467  3284#   4227*   4228*   4229
RESREG= 104410  4422    4463    4515    4575    5080    5106    5141    5203    5336    5414    5464    5511    5568
                5617    5690    5716    5753    5803    5910    5935    5995    6019    6118    6188    6320    6435
                6796    6845    6879    6940    8275    8327    8839#
RESVEC= 000010  1399#
RETAIL  033072  6454    6736    6800#
RETNML  033110  4601    4623    5862    6455    6803#
RKASOF= 000016  2410#   6822    7143    7213    7293    7548    7596    7785*   7810
RKBA  = 000004  2405#   6821    7544    7592    7818*
RKBADR  007517  3195#   3669
RKBAS   003036  2800#   3518*   3522*   3668    3671*   4768    6611    6641    6727    6782    6814    6996    7065
                7729
RKCS1 = 000000  2403#   4847*   4928*   4963*   4964    4966*   4967    5000*   6612*   6616    6642*   6646    6816
                6870*   6871    6873*   6874    7088    7140    7198    7248*   7257    7439*   7440    7449*   7488*
                7489    7513*   7541    7802*   7804    7843*   7855*   7857    7887*   7909*   7931*   7940*   7950
                7960    7975*
RKCS2 = 000010  2407#   3744*   3749    4818*   4823    6817    7066    7226    7484*   7542    7590    7773*   7839*
                7850*   7900*   7924*   7949*
RKDA  = 000006  2406#   6819    7243*   7545    7593    7779*   7821*   7923*
RKDB  = 000024  2413#   4970    4971    6613    6614    6615    6643    6644    6645    7222    7223    7224
RKDC  = 000020  2411#   6818
RKDCYL= 000020  2412#   7242*   7549    7598    7778*   7820*   7922*
RKDS  = 000012  2408#   6728    6823    7091    7546    7594
RKECPS= 000030  2418#   7157    7553
RKECPT= 000032  2420#   7156    7554
RKER  = 000014  2409#   6824    7100    7266    7547    7595
RKMR1 = 000026  2414#   7234*   7313*   7550    7627*   7634*   7641*   7648*   7749*
RKMR2 = 000034  2415#   7551    7861
RKMR3 = 000036  2416#   7552    7862
RKPAT = 000032  2419#
RKPOS = 000030  2417#
RKPRI   003042  2802#   3503    3527*   3678    3694*   6995    7719
RKPRTY  007560  3201#   3683
RKVADR  007540  3198#   3674
RKVEC   003040  2801#   3501    3523*   3524*   3673    3676*
RKWC  = 000002  2404#   6820    7543    7591    7819*
RLS   = 000010  2481#   7899
ROTBLD  022404  3867    4017    4149    5091#
RWBUF   052476  3827    3990*   3991    4060    4136    4874    4881    5354    5359    5405    5409    5486    5535
```

```
                    5679    5685    6009    9591#
R.ABNL  037310      7008    7118    7193    7210    7388    7408    7668#   7878
R.CONT  037334      7070    7108    7172    7181    7252    7274    7297    7303    7336    7376    7400    7445    7497
                    7520    7658    7676#   7892    7946    7955    7968
R.NORM  037322      7146    7239    7348    7672#   7815    7888    7976
SAVPRS  005444      2936#   6006    6013
SAVREG= 104407      4359    4432    4474    4527    5037    5091    5120    5159    5311    5404    5432    5480    5525
                    5563    5677    5701    5705    5729    5733    5748    5872    5919    5969    6004    6012    6107
                    6170    6223    6333    6385    6447    6813    6854    6920    8260    8301    8838#
SAVWRD  003150      2913#
SCLR   = 000040     2484#   7949
SCNDRV  020736      4816#   4956
SCOPER= 104411      8840#
SCOPE1  043716      8645#   8840
SCRACH  005434      2932#   3602*   3605*   3610    3746*   3747*   4820*   4821*   5272*   5274*   5275    5285*   5286*
                    5287    5529*   5571    5577*   5580    5737*   5758    5764*   5767
SCRLST  005546      2951#   3575    3583*   3589    3605    3717*   3720*   3721*   3739    3770*
SECLST  006430      3063#   5131    5178
SEEK   = 000117     2431#   3981    4073    4368    7766    7776
SELDRV= 000101      2424#   3751    4825    7905
SELF    011005      3322#   4240
SERNM   011517      3390#   5316    5358
SETTUP  021424      3791    3814    3854    3896    3941    3975    4047    4124    4221    4898#
SKI    = 000002     2500#   6568
SOFTBS  011545      3394#   5908
SPACE1  011720      3420#   4253
SPACE2  011717      3419#   4258    5889    5895    6262    6266    6269    6272    6292    6297    6312
SPACE3  011716      3418#
SPACE4  011715      3417#
SPACE6  011713      3416#
SPAR   = 020000     2464#   6518    7501
SPDLSS= 000020      2524#   6532
SRHTBS  027034      6068#   6109
SRPASS  010057      3236#   3701*   3702    3916*   3917
SRTSPL= 000111      2428#   7788
STACK  = 001100     1303#   3450    4898*   4899    4941    4948    5013    5020    5646    8629
STALL   023556      5378#   6152
STALLS  005422      2927#   3511*   5387
STDLST  006410      3053#   5039
STKLMT= 177774      1314#
STRCMD  027354      6156    6170#
STR204  010220      3253#   3909*   3910
STR220  010275      3261#   3919*   3920    4306*   4307
SUBCLR= 000177      2445#   4410    4772    4903    6783    6865    7943
SUBSYS  005500      2945#   3528*   3656*   3658    3729    3905    4302
SVAL   = 100000     2534#
SVPRMS  026634      3997    4408    6004#   6050
SWR     001140      1493#   3448    3469*   3471    3477*   3484*   4741    4982    5380    5578    5605    5765    5792
                    6225    6252    6280    8535    8542    8554    8557    8647    8667    8683
SWREG   000176      1419#   3477    4741    4743    4746*
SWRMSG  011015      3325#   4744
SW0    = 000001     1367#
SW00   = 000001     1357#   1367
SW01   = 000002     1356#   1366
SW02   = 000004     1355#   1365
SW03   = 000010     1354#   1364
```

```
SW04   = 000020          1353#   1363
SW05   = 000040          1352#   1362
SW06   = 000100          1351#   1361
SW07   = 000200          1350#   1360
SW08   = 000400          1349#   1359
SW09   = 001000          1348#   1358
SW1    = 000002          1366#
SW10   = 002000          1347#
SW11   = 004000          1346#
SW12   = 010000          1345#
SW13   = 020000          1344#   6225
SW14   = 040000          1343#
SW15   = 100000          1342#
SW2    = 000004          1365#
SW3    = 000010          1364#
SW4    = 000020          1363#
SW5    = 000040          1362#
SW6    = 000100          1361#
SW7    = 000200          1360#
SW8    = 000400          1359#
SW9    = 001000          1358#
SYSDRV   007653          3212#   3549*   3550
S.ACLO=  000100          2569#
S.BRHM=  000100          2584#
S.BRKE=  040000          2606#
S.CART=  000400          2586#
S.DCLO=  010000          2576#
S.DIB  = 002000          2602#
S.DOOR=  000200          2585#
S.DRA  = 000040          2555#
S.DROT=  020000          2577#
S.DRY  = 000200          2557#   4851
S.DSC  = 040000          2564#   7175    7323    7340    7403
S.FLT  = 000200          2570#   7318
S.FORM=  001000          2559#
S.FWD  = 002000          2588#
S.HDFL=  000200          2599#
S.HDHM=  000040          2583#
S.ICYL=  000040          2568#
S.ILF  = 000400          2571#
S.LIMD=  020000          2605#
S.LOAD=  010000          2590#
S.MHD  = 000400          2600#
S.NMOV=  010000          2604#
S.OFF  = 002000          2560#
S.PAR  = 001000          2572#   7178    7655
S.PIP  = 020000          2563#   7186    7381
S.PLO  = 004000          2603#
S.REV  = 004000          2589#
S.RTZ  = 020000          2591#
S.SECT=  000020          2596#
S.SKI  = 002000          2573#
S.SPIN=  010000          2562#
S.SPLS=  010000          2575#
S.SPOK=  001000          2587#
S.TYPE=  000400          2558#
```

```
S.UNLD= 040000          2592#
S.UNS = 040000          2578#
S.VV  = 000100          2556#
S.WCLK= 000040          2597#
S.WGAT= 000100          2598#
S.WLE = 004000          2574#
S.WRL = 004000          2561#  4859
S.XDOK= 000020          2582#
S.XERR= 001000          2601#
TAB     011013          3324#  4262
TABLEA  005556          2959#  3824    3826
TABLEB  005656          2980#  5406
TABLEC  005716          3001#  3859    3862    3870
TABLED  005750          3012#  5040    5055    5069    5072    5092    5104    5124    5176
TABLEG  006350          3032#  3989    5123
TBITVE= 000014          1400#
TKVEC = 000060          1407#  3499*   3500*
TOPROC  033120          6336   6503    6812#
TPVEC = 000064          1408#
TRACK   005430          2930#
TRAPVE= 000034          1406#  3456*   3457*   8628*
TRKAVG  017710          4276   4527#
TRKBUF  010774          3320#  4233*   4236
TRNSFR  026724          3998   4409    6033#
TRTVEC= 000014          1401#
TSTDRN  011454          3382#
TSTING  003117          2879#  3504*   3817*   3899*   3978*   4224*   4296*   5632    5644*
TSTMSG  044534          6246   8842#
TSTTYP  003133          2891#  3489*   4003*   4022*   4071*   4078*   4091*   4127*   4155*   4361    4380    4433    4450
                        4475   4485    5160    5166    5649*   6618    6663    6720    6740
TST1    014076          3705   3789#   3908
TST10   015444          4122#
TST11   016062          4219#
TST2    014126          3812#
TST3    014270          3852#
TST4    014410          3894#
TST5    014574          3707   3918    3939#   4305
TST6    014632          3973#
TST7    015136          4045#
TWOTOS= 002000          3180#  6504    6833
TYPBSF  026164          4984   5872#
TYPOS = 104405          8834#
TYPE  = 104401          3494   3497    3539    3550    3559    3560    3595    3601    3609    3610    3616    3624    3636
                        3637   3642    3651    3652    3663    3669    3674    3683    3695    3697    3698    3702    3910
                        3917   3920    4229    4236    4240    4245    4253    4256    4258    4262    4267    4270    4283
                        4284   4289    4294    4307    4310    4642    4648    4652    4656    4658    4659    4690    4691
                        4709   4719    4720    4744    4844    4845    4855    4856    4863    4864    4880    4885    4886
                        4929   4930    4931    4938    4939    5001    5002    5003    5010    5011    5242    5248    5254
                        5260   5271    5275    5279    5287    5292    5294    5295    5297    5315    5316    5335    5357
                        5358   5362    5658    5659    5875    5876    5877    5878    5884    5885    5889    5895    5899
                        5908   6239    6240    6241    6245    6246    6251    6254    6255    6256    6257    6262    6265
                        6266   6267    6268    6269    6272    6275    6276    6278    6292    6297    6301    6307    6312
                        6313   6315    8121    8368    8433    8508    8537    8545    8630    8830#
TYPERR  027506          6223#  8544
TYPFMT  003123          2883#
TYPOC = 104402          5888   5894    5898    6261    6271    6274    6294    8831#
```

```
TYPON = 104404          8833#
TYPOS = 104403          4706    6248    8832#
TYPRWN  010352          3269#   4930    5002
TYPSYS  007600          3204#   3642
T.ASOF  003014          2787#   7143*   7144    7168    7213*   7214    7293*   7294    7300    7331    7365    7373    7397
                        7548*   7810*   7811
T.BA    003006          2784#   7362    7544*
T.CS1   003000          2779#   7088*   7140*   7141    7154    7165*   7198*   7200    7206    7235*   7257*   7259    7289
                        7314*   7328*   7358*   7359    7370*   7394*   7440*   7441    7481    7485*   7486*   7488    7492
                        7501    7541*   7588*   7589    7629*   7636*   7643*   7650*   7804*   7805    7807
T.CS2   003002          2781#   7066*   7067    7082    7085    7097    7126    7226*   7227    7263    7277    7360    7482*
                        7483*   7484    7494    7503    7542*
T.DA    003010          2785#   7363    7545*
T.DB    003034          2796#
T.DC    003012          2786#   7364    7549*
T.DS    003020          2790#   7091*   7092    7367    7546*
T.ER    003016          2789#   7100*   7103    7266*   7269    7366    7505    7547*
T.MR1   003022          2791#   7550*
T.MR2   003024          2792#   7175    7186    7322    7339    7381    7403    7551*   7632    7639    7646    7653
T.MR3   003026          2793#   7178    7238    7317    7552*   7633    7640    7647    7654    7655
T.PAT   003032          2795#   7554*
T.POS   003030          2794#   7553*
T.WCR   003004          2783#   7361    7543*
UEXATT= 000010          2671#   6710    7149
UFE     = 000400        2487#   6487    6493    7082    7160    7503    7873
UNLOAD= 000107          2427#   7754
UNS     = 040000        2515#   6563
UPE     = 020000        2492#   6475    7097    7263    7503
VFHDSW= 000002          2906#
VV      = 000100        2527#
WCDASW= 000004          2907#
WCE     = 040000        2493#   6686    7097    7263    7503
WCEFLG  003135          2893#   5828*   5830    5845*   5847    5861*
WCERR   = 000100        3176#   6689    6766
WDSXFR  005456          2937#   5481    5528    5560    5707    5735    5745    5934*   5974    6007*   6008*
WHDSW   = 000001        2905#
WILTST  007700          3216#   3601
WLE     = 004000        2512#   6559    7103    7269
WORDSP  011726          3423#
WRDATA= 000123          2433#   3830    3985    4076    5688    7826
WRHEAD= 000127          2435#
WRL     = 004000        2531#
WRTBLK  022470          3860    3863    3871    5120#
WRTCHK= 000131          2436#   3833    7822
WRTGAT= 040000          2550#
WRTLOK  011117          3340#   4864
WRTSEC  025126          5133    5677#
W.DRV   003114          2876#   6997*   7112*   7138*   7190*   7205*   7218*   7232*   7280*   7347*   7356*   7385*   7407*
                        7678*   7727*   7790*   7793*   7814*   7866*   7884*
W.MILI  003060          2829#   6992    7958
W.MIN   003066          2845#   7790
W.MTIM  003056          2828#   6990*   6992*
W.SEC   003062          2842#   7727
W.TIME  003100          2851#   6993    6999*   7111*   7139*   7189*   7204*   7217*   7233*   7279*   7346*   7355*   7384*
                        7406*   7677*   7726*   7813*   7865*   7883*
W.WCK = 000200          2676#   7737    7798    7824    7830
```

```
W.WTCH  033726          6160   6985#  7803   7809   7856
W.8SEC  003064          2844#  7190   7385   7793
$APTHD  001000          1450   1456#
$ASTAT= ****** U        8731   8746
$ATYC   044136          8702   8704#
$ATY1   044112          8700#
$ATY3   044120          8106   8701#
$ATY4   044130          8550   8703#
$AUTOB  001134          1490#
$BASE   001404          1601#  3518
$BOAOR  001122          1485#
$BODAT  001126          1487#
$BELL   001320          1552#  8537   8569
$CHARC  041766          8123*  8133*  8140   8149*  8154#
$CKSWR= ****** U        8837
$CMTAG  001100          1473#  3445   3446   3454   3460   3461
$CM1  = 000040          1505#  1506#  1507#  1508#  1509#  1510#  1511#  1512#  1513#  1514#  1515#  1516#  1517#
                        1518#  1519#  1520#  1521#  1522#  1523#  1524#  1525#  1526#  1527#  1528#  1529#  1530#
                        1531#  1532#  1533#  1534#  1535#  1536#  1537#
$CM2  = 000100          1505#  1506#  1507#  1508#  1509#  1510#  1511#  1512#  1513#  1514#  1515#  1516#  1517#
                        1518#  1519#  1520#  1521#  1522#  1523#  1524#  1525#  1526#  1527#  1528#  1529#  1530#
                        1531#  1532#  1533#  1534#  1535#  1536#  1537#
$CM3  = 000040          1503#  1505
$CM4  = 000016          1537#  1538#  1539#  1540#  1541#  1542#  1543#  1544#  1545#  1546#  1547#  1548#  1549#
                        1550#  1551#
$CNTLG  043557          8607#
$CNTLU  043552          8606#
$CPUOP  001356          1575#
$CRLF   001325          1554#  3539   3695   4267   4270   4289   4294   4659   5292   5297   5335   5362   5878
                        5885   5899   6245   6255   6257   6265   6268   6278   6301   6307   6315   8122   8157
                        8545   8569
$DBLK   043226          8474   8508   8516#
$DB2D   042330          4260   4287   8301#
$DB2O   042210          5360   8260#
$DECVL  042510          8303   8349#
$DEVCT  001340          1566#
$DEVM   001406          1602#  3712
$DOAGN  016722          4334   4339   4345#
$DTBL   043216          8477   8512#
$ENDAD  016712          1436   4341#  8564
$ENDCT  016676          3460   4336#
$ENV    001350          1571#  8101   8547   8709   8733
$ENVM   001351          1572#  3482   3519   3713   3726   8103   8108   8711
$EOP    016646          4327#
$EOPCT  016670          3460*  4333#  4337
$ERFLG  001103          1476#  8532*  8569   8645   8659   8681   8687*  8697
$ERMAX  001115          1482#  3462*  8693*  8697
$ERROR  043236          3454   8531#
$ERRPC  001116          1483#  8539*  8540*  8541   8569   9312
$ERRTB  001410          1619#  6236
$ERTTL  001112          1480#  8538*  8569
$ESCAP  001316          1551#  3461*  8560   8562   8569   8692*
$ETABL  001350          1570#
$ETEND  001410          1462   1603#
$FATAL  001332          1563#  8737*
$FFLG   044356          8700*  8703*  8731   8740*  8748#
```

```
$FILLC  001156          1501#  8126   8157
$FILLS  001155          1500#  8157
$GDADR  001120          1484#
$GDDAT  001124          1486#
$GET42  016702          4338#
$GTSWR= ******  U       8836
$HD     =  000002       1284
$HIBTS  001000          1457#
$HINUM  041504          3513*  8058   8066   8071*  8076#
$HIOCT  041404          4727   8028*  8041#
$ICNT   001104          1477#
$INTAG  001135          1491#
$ITEMB  001114          1481#  6231   8541*  8549   8569
$LF     001326          1555#  8157   8569
$LFLG   044355          8741*  8747#
$LONUM  041506          3514*  5383   8057   8064   8070*  8077#
$LPADR  001106          1478#  3463*  8632   8685*  8690*  8695   8697
$LPERR  001110          1479#  3464*  8559   8649   8685   8691*  8697
$MADR1  001362          1588#
$MADR2  001366          1592#
$MADR3  001372          1595#
$MADR4  001376          1598#
$MAIL   001330          1458   1462   1561#  3481   3790   3813   3853   3895   3940   3974   4046   4123   4220
                        8101   8547   8689
$MAMS1  001360          1582#
$MAMS2  001364          1590#
$MAMS3  001370          1593#
$MAMS4  001374          1596#
$MBADR  001002          1458#
$MFLG   044354          8701*  8707   8742*  8746#
$MNEW   043575          8610#
$MSGAD  001344          1568#  8717*  8720
$MSGLG  001346          1569#  8722*
$MSGTY  001330          1562#  8715   8723*  8735   8739*
$MSWR   043564          8608#
$MTYP1  001361          1583#
$MTYP2  001365          1591#
$MTYP3  001371          1594#
$MTYP4  001375          1597#
$NULL   001154          1499#  8128   8157
$NWTST= 000001          3777#  3779   3800#  3802   3843#  3845   3875#  3877   3926#  3928   3949#  3951   4026#
                        4028   4095#  4097   4199#  4201
$OCNT   043006          8405*  8434*  8447#
$OCTVL  042312          5318   8262   8287#
$OMODE  043010          8400*  8404*  8409   8412*  8423*  8449#
$OVER   044076          8668   8686   8694#
$PASS   001336          1565#  3481*  4330*  4331*  4347
$PASTM  001006          1460#
$PWRCT  043672          8623*  8624*  8633#
$PWRON  043606          3458   8618#  8626
$PWRUP  043620          8618   8623#
$QUES   001324          1553#  3560   3637   3652   3698   4691   4720   4939   5011   5295   8157   8569
$RAND   041406          5382   8053#
$RDCHR  043432          8585#  8837
$RDDEC= ******  U       8838
$RDLIN= ******  U       8838
```

```
$RDOCT= ****** U        8838
$RDSZ = 000000          8606#
$REGAD  001160          1503#
$REG0   001162          1505#   6388    9312
$REG1   001164          1506#   9312
$REG10  001202          1513#   5559*   5560*   5561    5592*   5595    5744*   5745*   5746    5779*   5782    6613*   6643*
                        6682*   9316    9324
$REG11  001204          1514#   5593*   5780*   6614*   6644*   6683*   9316    9324
$REG12  001206          1515#   5594*   5781*   6615*   6645*   6676*   6678*   9316    9324
$REG13  001210          1516#   5596*   5598*   5600*   5601*   5783*   5785*   5787*   5788*   9316    9324
$REG14  001212          1517#   5595*   5597*   5599*   5782*   5784*   5786*   9319    9324
$REG15  001214          1518#   5602*   5603*   5789*   5790*   9319    9324
$REG16  001216          1519#   9320    9324
$REG17  001220          1520#   9320
$REG2   001166          1507#   9312
$REG20  001222          1521#   9320
$REG21  001224          1522#   9320
$REG22  001226          1523#   9320
$REG23  001230          1524#   9320
$REG24  001232          1525#   9320
$REG25  001234          1526#   9320
$REG26  001236          1527#   6259    6419
$REG27  001240          1528#
$REG3   001170          1508#   9312
$REG30  001242          1529#
$REG31  001244          1530#
$REG32  001246          1531#
$REG33  001250          1532#
$REG34  001252          1533#
$REG35  001254          1534#
$REG36  001256          1535#   6398*   9315
$REG37  001260          1536#   6399*   9315
$REG4   001172          1509#
$REG5   001174          1510#   5170    5574*   5582    5588*   5761*   5769    5775*   6207*   6208*   6395    6777*   6778*
                        6815    6887*   6909*   6921*   6935    6938    9316    9323
$REG6   001176          1511#   5193    5196    5575*   5584    5589*   5762*   5771    5776*   6888*   6891*   6904    6906*
                        6908*   6928*   6929*   6930*   6931*   6934*   6936    6937    9316    9323
$REG7   001200          1512#   5178    5576*   5586    5590*   5763*   5773    5777*   6889*   6890*   6894    6897*   6901*
                        6903*   6935*   6937*   6939*   9316    9323
$RESRE  044416          8787#   8839
$R2A  = ****** U        8840
$SAVRE  044360          8771#   8838
$SCOPE  043742          3452    8666#
$SETUP= 000037          3429#   3451    3452    3454    3456    3458    3460    3461    3463    4329    8532    8557    8564
                        8574    8612    8667
$STUP = 177777          3429#
$SUPRS  042524          4261    4288    5334    5361    8360#
$SVLAD  044042          8676    8682    8688#
$SVPC = 000230          1434#   1439
$SWR  = 163000          1284#   1286#   1291    1292    1293    1294    1295    1296    1297    1551    1552    3461    3463
                        3464    3790    3813    3853    3895    3940    3974    4046    4123    4220    4324    4330    4340
                        4346    4347    8523    8524    8525    8526    8527    8535    8542    8554    8557    8569    8660
                        8661    8662    8663    8667    8679    8681    8682    8683    8688    8691    8694    8697
$SWREG  001352          1573#   3484
$SWRMK= 000000          8663
$TESTN  001334          1564#   3790*   3813*   3853*   3895*   3940*   3974*   4046*   4123*   4220*   8689*
```

```
STKB    001146          1496#  4632   4674#  8572   8589   8595
STKS    001144          1495#  4643#  4676#  8572   8587   8593
STMP0   001262          1537#
STMP1   001264          1538#  6072#  6073#  6081
STMP10  001302          1545#  6085#  6086#  6087
STMP11  001304          1546#  4160#  4162#  4166   4180#  4182#  4186   4250#  4265#  4280#  4292#  4477#  4482#
                        4487#  4496   4508#  4509#  4559#  4564   4571#  4572#  5095#  5101
STMP12  001306          1547#  4166#  4167#  4168   4170#  4171   4186#  4187#  4188   4190#  4191   4230#  4234
                        4272#  4480#  4490#  4513#  4560#  4573#  5096#  5102
STMP13  001310          1548#  4248#  4263#  4264   4478#  4483#  4488#  4499   4511#  4512#  5097#  5103
STMP14  001312          1549#
STMP15  001314          1550#
STMP2   001266          1539#  6068#  6096
STMP3   001270          1540#  6069#  6097
STMP4   001272          1541#  6070#  6098
STMP5   001274          1542#
STMP6   001276          1543#
STMP7   001300          1544#
STN   = 000012          1273#  1284   3777   3790#  3800   3813#  3843   3853#  3875   3895#  3926   3940#  3949
                        3974#  4026   4046#  4095   4123#  4199   4220#
STNPWR  042440          8308   8309   8329#
STPB    001152          1498#  8146#  8157
STPFLG  001157          1502#  8095   8157
STPS    001150          1497#  8144   8157
STRAP   044454          3456   8807#
STRAP2  044476          8818#  8829
STRAP   000012          8822#  8831#  8832#  8833#  8834#  8835#  8837   8838#  8839#  8840#  8841#
STRPAD  044510          8812   8829#
STSTM   001004          1459#
STSTMM  001102          1475#  3492#  3540#  3641#  3706#  3907#  4304#  4329#  5650#  6247   8534   8569   8659
                        8688#  8689   8694   8697
STYPBN= ****** U        8835
STYPDS  043012          8462#  8834
STYPE   041510          8095#  8728   8822   8830
STYPEC  041722          8125   8132   8139   8144#  8145
STYPEX  041770          8150   8152   8155#
STYPOC  042610          8403#  8831
STYPON  042624          8402   8405#  8833
STYPOS  042564          8398#  8832
SUNIT   001342          1567#
SUNITM  001010          1461#
SUSWR   001354          1574#
SVECT1  001400          1599#  3517
SVECT2  001402          1600#
SXTSTR  043752          8670#
$$GET4= 000000          4340#
S.S.SM= ****** U        2828   7014   7249   7440   7451   7489   7803   7844   7856   7910   7932   7950   7977
                        7986
SOFILL  043007          8399#  8403#  8413   8448#
$40CAT= ****** U        8544   8667
.     = 054236          1413#  1417#  1422#  1424#  1434   1435#  1437#  1439#  1440#  1446   1447#  1449#  1451#
                        1472#  1556   2914#  2915#  2920#  2936#  2943#  2944#  2946#  2951#  3012#  3013#  3014#
                        3015#  3016#  3017#  3018#  3019#  3020#  3021#  3022#  3023#  3024#  3025#  3026#  3027#
                        3073#  3076#  3079#  3084#  3087#  3090#  3095#  3098#  3101#  3106#  3109#  3112#  3148#
                        3151#  3154#  3159#  3162#  3165#  3449   3463   3464   4347   8157   8229   8287#  8349#
                        8516#  8569   8572   8612   8620   8637#  8697   8749#  9591#
```

```
.$ASTA= ****** U        8701    8704
.$X   = 001000          1446#   1451
..HIGH  000226          1428#
..LOW   000224          1427#
```

```
CKEXIT  2128#  6153
COMMEN  1410#
ENDCOM  1410#
ERROR   1304#  3755  4829  5573  5591  5604  5760  5778  5791  5836  6209  6339  6343  6350  6354
        6358   6362  6366  6369  6372  6375  6376  6477  6482  6489  6491  6497  6499  6508  6510
        6512   6516  6520  6525  6530  6534  6538  6541  6545  6549  6553  6557  6561  6565  6570
        6575   6580  6585  6590  6595  6605  6622  6625  6635  6648  6650  6667  6684  6693  6697
        6702   6707  6712  6746  6748  6779
ESCAPE  1410#
GETPRI  1410#
GETSWR  1410#
MSG     3777#  3779  3800#  3802  3843#  3845  3875#  3877  3926#  3928  3949#  3951  4026#  4028  4095#
        4097   4199#  4201
MULT    1410#
NEWTST  1410#  3777  3800  3843  3875  3926  3949  4026  4095  4199
NEXTAD  2124#
NXTAD   2120#
PARMBK     1#  2712  2744
PARMDF     1#  2618
POP     1410#  8072  8503  8743  8744  8792
PUSH    1410#  8053  8462  8704  8706  8727  8772
P.DRVR     1#  2776
REGDEF     1#  2400
REPORT  1410#
R.DRIV     1#  6943
R.QDRV     1#
SCOPE   1305#  3789  3812  3852  3894  3939  3973  4045  4122  4219  4328
SETPRI  1410#
SETTRA  8822#  8831  8832  8833  8834  8837  8838  8839  8840
SETUP   1410#  3444
SKIP    1410#
SLASH   1410#
SPACE   1410#
STARS   1262   1269  1410#  1432  1443  1445  1452  1468  1556  1559  3533  3537  3777  3788  3800
        3811   3843  3851  3875  3893  3926  3938  3949  3972  4026  4044  4095  4121  4199  4218
        4322   4350  4358  4427  4431  4468  4473  4520  4526  4580  4585  4607  4612  4627  4629
        4664   4672  4681  4688  4699  4704  4734  4740  4756  4764  4781  4786  4801  4814  4893
        4897   4912  4919  4990  4994  5029  5036  5085  5090  5111  5119  5146  5158  5208  5229
        5306   5310  5341  5345  5370  5376  5400  5403  5422  5431  5469  5478  5516  5524  5622
        5631   5671  5676  5695  5700  5721  5728  5810  5813  5866  5871  5915  5918  5940  5944
        5962   5968  6000  6003  6024  6032  6057  6067  6100  6106  6134  6150  6167  6169  6193
        6199   6213  6222  6322  6332  6379  6437  6445  6807  6811  6850  6852  6884  6886  6914
        6919   6952  6983  7017  7057  7420  7437  7455  7479  7526  7539  7558  7584  7603  7625
        7683   7710  7988  8001  8044  8080  8160  8170  8188  8199  8251  8290  8352  8375  8452
        8519   8571  8577  8641  8644  8656  8699  8756  8801
SWRSU   1259#  1410#  3465#
S.DRVE     1#  2552
TEST    2116#  3777  3800  3843  3875  3926  3949  4026  4095  4199
TRMTRP  8822#
TYPBIN  1410#
TYPDEC  1410#
TYPNAM  1410#
TYPNUM  1410#
TYPOCS  1410#
TYPOCT  1410#
TYPTXT  1410#
```

```
$DECBN      1#
$OCTBN      1#  7986
$SCMRE   1466#  1505  1506  1507  1508  1509  1510  1511  1512  1513  1514  1515  1516  1517  1518
         1519   1520  1521  1522  1523  1524  1525  1526  1527  1528  1529  1530  1531  1532  1533
         1534   1535  1536
$SCMTM   1466#  1537  1538  1539  1540  1541  1542  1543  1544  1545  1546  1547  1548  1549  1550
$SESCA   1410#
$SNEWT   1410#  3777  3800  3843  3875  3926  3949  4026  4095  4199
$SSET    8822#  8831  8832  8833  8834  8837  8838  8839  8840
$SSETM   3481#
$SSKIP   1410#
.EQUAT   1259#  1300
.HEADE   1259#  1274
.SETUP   1259#  3429
.SWRHI   1259#  1287
.SWRLO   1259#  1298#
.$ACTI   1259#  1430
.$APTB   1557#
.$APTH   1259#  1441
.$APTY   1259#  8697
.$CATC   1259#  1411
.$CMTA   1259#  1466
.$D82D   1259#  8288
.$D820   1259#  8249
.$EOP    1259#  4320
.$ERRO   1259#  8517
.$POWE   1259#
.$RAND   1259#  8042
.$READ   1259#  8569
.$SAVE   1259#  8754
.$SCOP   1259#  8654
.$SIZE   1259#
.$SUPR   1259#  8350
.$TRAP   1259#  8799
.$TYPD   1259#  8450
.$TYPE   1259#  8078
.$TYPO   1259#  8373

. ABS.  054236     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZR6QA,DZR6QA.SEQ/SOL/CRF/NL:TOC/DOC=DRIV10.P11/EQ:QNEWSW,DZR6QA.P11
 RUN-TIME: 31 27 2 SECONDS
 RUN-TIME RATIO: 218/61=3.5
 CORE USED:  48K  (96 PAGES)

 DOCUMENT PAGES:  212
```