RK611/RK06

DRIVE COMPATIBILITY PROGRAM
CZR6QB0

AH-B162B-MC

FICHE 1 OF 2

MAR 1978

digital

MADE IN USA

RK611/RK06

DRIVE COMPATIBILITY PROGRAM
CZR6QB0

AH-B162B-MC

FICHE 2 OF 2

MAR-1978
digital
MADE IN USA

```
                                .REM @
```

IDENTIFICATION

PRODUCT CODE:	AC-B161B-MC
PRODUCT NAME:	CZR6QB0 RK6 DR CPT PROG
DATE:	FEB 1978
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DAVE HOFFMAN

TABLE OF CONTENTS

103
104
105
106
107
108
109
110
111
112
113
114

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 OR RK07 DRIVES WHICH MAY RESIDE ON ONE OR MORE RK611/RK06-07 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.

NOTE: RK06 AND RK07 DRIVES CANNOT BE MIXED FOR COMPATIBILITY TESTING.

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 RK06-07 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 OR RK07 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 CZR6A-CZR6E AND CZR6K, THE DRIVE DIAGNOSTICS CZR6H-CZR6J, AND THE SUBSYSTEM VERIFICATION PROGRAMS CZR6M-CZR6N. IN ADDITION, THE CARTRIDGE MUST BE FORMATTED IN 20 OR 22 SECTOR FORMAT USING THE PACK FORMATTER CZR6L, 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:

CZR6QB0 - RK06-RK07 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:

DRIVE TYPE 6<CR> FOR RK06, 7<CR> FOR RK07:

THE OPERATOR MUST TYPE EITHER A 6<CR> OR A 7<CR>

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:

CZR6QB0 - RK06-RK07 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) :

DRIVE TYPE 6<CR> FOR RK06, 7<CR> FOR RK07:

THE OPERATOR MUST TYPE IN 6<CR> OR 7<CR>

SUBSYS A DRIVE(S) =

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

```
437                                   SUBSYS A DRIVE(S) = 2,5,7
438
439                                THE PROGRAM THEN VERIFIES THE DRIVE NOS.  BY TYPING :
440
441                                   WILL TEST DRIVE(S) 2,5,7 ON SUBSYS A.
442
443                                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 :

    CZR6QB0 - RK06-RK07 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 **

```
612
613
614
615                    (2) IF THERE IS ANOTHER SUBSYSTEM, HOWEVER, THE  PROGRAM  DIRECTS  THE
616                    OPERATOR  TO PERFORM PASS 2 ON THE NEXT SUBSYSTEM AS FOLLOWS (AND THEN
617                    HALTS):
618
619                         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

870
871
872
873
874

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

```
                                22.  DATA LATE ERROR
                                23.  CONTROLLER TIMEOUT ERROR
                                24.  OPERATION INCOMPLETE ERROR
                                25.  HEADER VRC ERROR
                                26.  DATA CHECK ERROR
                                27.  WRITE CHECK ERROR
                                28.  DATA MISCOMPARE
                                29.  NO DRIVE RESPONSE - UFE AND NXD
                                30.  DRIVE ERROR WILL NOT CLEAR
                                31.  DRIVE STATUS CHANGE WILL NOT CLEAR
                                32.  ATTENTION BUT NO STATUS CHANGE OR FAULT
                                33.  ATTENTION BUT DRIVE NOT AVAILABLE
                                34.  ERROR WHILE GATHERING DRIVE STATUS
                                35.  MULTIPLE DRIVE SELECT
                                36.  HEADER COMPARE ERROR
                                37.  ERROR IN RECALIBRATE FOR RECOVERY
                                38.  CLEAR CONTROLLER DID NOT CLEAR ERROR
                                39.  NO ATTENTION IN ATTENTION SUMMARY REGISTER
                                40.  UNSOLICITED ATTENTION
                                41.  UNEXPECTED DATA TYPE ERROR
                                42.  ATTENTION DID NOT RESET WITH CLEAR
                                43.  SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
                                44.  DATA LATE WHEN UNLOADING HEADER
                                45.  CONTROLLER ERROR WHEN DRIVER SERVICING
                                46.  RETRY UNSUCCESSFUL
                                47.  BAD SECTOR ERROR ON SECTOR NOT LISTED BAD
```

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 RK06/RK07	SECTOR
0	620/1444	0
1	620/1444	2
2	620/1444	4
3	620/1444	6
4	620/1444	10
5	620/1444	12
6	620/1444	14
7	620/1444	16
10	622/1446	0
11	622/1446	2
12	622/1446	4
13	622/1446	6
14	622/1446	10
15	622/1446	12
16	622/1446	14
17	622/1446	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

1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130

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	RK06
2 - 100-177	100-117	160-177	3	RK06
3 - 200-277	200-217	260-277	7	RK06
4 - 300-377	300-317	360-377	13	RK06
5 - 400-477	400-417	460-477	17	RK06
6 - 500-577	500-517	560-577	22	RK06
7 - 600-632	600-617	---	0	RK06
1 - CYL 0-177	CYL 0-17	CYL 160-177	SECT 0	RK07
2 - 200-377	200-217	360-377	3	RK07
3 - 400-577	400-417	560-577	7	RK07
4 - 600-777	600-617	760-777	13	RK07
5 - 1000-1177	1000-1017	1160-1177	17	RK07
6 - 1200-1377	1200-1217	1360-1377	22	RK07
7 - 1400-1456	1400-1417	---	0	RK07

TABLE D

BASIC CYLINDER BLOCK LAYOUT EXAMPLE

(SHOWN FOR RK06, USE CYL 160-177 FOR RK07)

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

NOTE: THE ABOVE TABLE GIVES CYLINDER NUMBERS FOR THE RK06.
REFER FOR TABLE C TO OBTAIN CORRESPONDING CYLINDER
NUMBERS FOR AN RK07.

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

NOTE: THE ABOVE TABLE GIVES CYLNDER NULMBERS FOR THE RK06.
REFER TO TABLE C TO OBTAIN CORRESPONDING CYLINDER
NUMBERS FOR THE RK07.

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

                               ;*** REV 003 ***




                               ;;*****************************************************************
                               .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  CZR6QB0 RK6 DR CPT PROG
                               ;*COPYRIGHT (C) 1977
                               ;*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
```

```
                        
                        HT=     11              ;;CODE FOR HORIZONTAL TAB
                        LF=     12              ;;CODE FOR LINE FEED
                        CR=     15              ;;CODE FOR CARRIAGE RETURN
                        CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
                        PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
                        STKLMT= 177774          ;;STACK LIMIT REGISTER
                        PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
                        DSWR=   177570          ;;HARDWARE SWITCH REGISTER
                        DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                        
                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
                        R0=     %0              ;;GENERAL REGISTER
                        R1=     %1              ;;GENERAL REGISTER
                        R2=     %2              ;;GENERAL REGISTER
                        R3=     %3              ;;GENERAL REGISTER
                        R4=     %4              ;;GENERAL REGISTER
                        R5=     %5              ;;GENERAL REGISTER
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER
                        
                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7
                        
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
```

```
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
            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
```

STARTING ADDRESS(ES)

```
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  012176                  JMP     @#DFSTRT        ;;JUMP TO STARTING ADDRESS OF PROGRAM
        000204                          .=204
000204  000137  012204                  JMP     @#P1STRT
        000220                          .=220
000220  000137  012222                  JMP     @#P2STRT

000224  000700                  ..LOW:  .WORD   700
000226  001100                  ..HIGH: .WORD   1100

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000230                          $SVPC=.                 ;SAVE PC
        000046                          .=46
000046  017264                  $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  140000                  .WORD   140000                  ;;2)SET LOC.52 TO 140000
        000230                          .=$SVPC                 ;; RESTORE PC
        001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
        001000                          .$X=.   ;;SAVE CURRENT LOCATION
        000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                  $APTHDR ;;POINT TO APT HEADER BLOCK
        001000                          .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001330                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  001320                  $TSTM:  .WORD   1320    ;;RUN TIM OF LONGEST TEST
001006  001546                  $PASTM: .WORD   1546    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  001546                  $UNITM: .WORD   1546    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000030                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
        120210                  AVECT1=120210           ;RKVEC=210, RKPRI=5
        177440                  ABASE=177440            ;RKBAS ADRS
        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)
```

```
1555  001224  000000                  $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
1556  001226  000000                  $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
1557  001230  000000                  $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
1558  001232  000000                  $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
1559  001234  000000                  $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
1560  001236  000000                  $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
1561  001240  000000                  $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
1562  001242  000000                  $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
1563  001244  000000                  $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
1564  001246  000000                  $REG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
1565  001250  000000                  $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
1566  001252  000000                  $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
1567  001254  000000                  $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
1568  001256  000000                  $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
1569  001260  000000                  $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
1570  001262  000000                  $TMP0:  .WORD   0               ;;USER DEFINED
1571  001264  000000                  $TMP1:  .WORD   0               ;;USER DEFINED
1572  001266  000000                  $TMP2:  .WORD   0               ;;USER DEFINED
1573  001270  000000                  $TMP3:  .WORD   0               ;;USER DEFINED
1574  001272  000000                  $TMP4:  .WORD   0               ;;USER DEFINED
1575  001274  000000                  $TMP5:  .WORD   0               ;;USER DEFINED
1576  001276  000000                  $TMP6:  .WORD   0               ;;USER DEFINED
1577  001300  000000                  $TMP7:  .WORD   0               ;;USER DEFINED
1578  001302  000000                  $TMP10: .WORD   0               ;;USER DEFINED
1579  001304  000000                  $TMP11: .WORD   0               ;;USER DEFINED
1580  001306  000000                  $TMP12: .WORD   0               ;;USER DEFINED
1581  001310  000000                  $TMP13: .WORD   0               ;;USER DEFINED
1582  001312  000000                  $TMP14: .WORD   0               ;;USER DEFINED
1583  001314  000000                  $TMP15: .WORD   0               ;;USER DEFINED
1584  001316  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
1585  001320  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
1586  001324     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
1587  001325     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1588  001326  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
1589                                  ;;**************************************************************
1590                                  .SBTTL  APT MAILBOX-ETABLE
1591
1592                                  ;;**************************************************************
1593                                  .EVEN
1594  001330                          $MAIL:                          ;;APT MAILBOX
1595  001330  000000                  $MSGTY: .WORD   AMSGTY           ;;MESSAGE TYPE CODE
1596  001332  000000                  $FATAL: .WORD   AFATAL           ;;FATAL ERROR NUMBER
1597  001334  000000                  $TESTN: .WORD   ATESTN           ;;TEST NUMBER
1598  001336  000000                  $PASS:  .WORD   APASS            ;;PASS COUNT
1599  001340  000000                  $DEVCT: .WORD   ADEVCT           ;;DEVICE COUNT
1600  001342  000000                  $UNIT:  .WORD   AUNIT            ;;I/O UNIT NUMBER
1601  001344  000000                  $MSGAD: .WORD   AMSGAD           ;;MESSAGE ADDRESS
1602  001346  000000                  $MSGLG: .WORD   AMSGLG           ;;MESSAGE LENGTH
1603  001350                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
1604  001350     000                  $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1605  001351     000                  $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1606  001352  000000                  $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1607  001354  000000                  $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1608  001356  000000                  $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1609                                  ;*                          BITS 15-11=CPU TYPE
1610                                  ;*                                  11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
```

```
1611                                    ;*                      11/70=06,PDQ=07,Q=10
1612                                    ;*                  BIT 10=REAL TIME CLOCK
1613                                    ;*                  BIT  9=FLOATING POINT PROCESSOR
1614                                    ;*                  BIT  8=MEMORY MANAGEMENT
1615  001360      000                   $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1616  001361      000                   $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1617                                    ;*                  MEM.TYPE BYTE   --  (HIGH BYTE)
1618                                    ;*                          900 NSEC CORE=001
1619                                    ;*                          300 NSEC BIPOLAR=002
1620                                    ;*                          500 NSEC MOS=003
1621  001362  000000                    $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1622                                    ;*                    MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1623  001364      000                   $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1624  001365      000                   $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1625  001366  000000                    $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1626  001370      000                   $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.JYTE
1627  001371      000                   $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1628  001372  000000                    $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1629  001374      000                   $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1630  001375      000                   $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1631  001376  000000                    $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1632  001400  120210                    $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1633  001402  000000                    $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1634  001404  177440                    $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1635  001406  000377                    $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1636  001410                            $ETEND:
1637                                    .MEXIT
```

```
                                ERROR POINTER TABLE

                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $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  045417                         EM1
001412  050512                         DH100
001414  052276                         DT100
001416  052406                         DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
001420  045443                         EM2
001422  050512                         DH100
001424  052276                         DT100
001426  052406                         DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
001430  045467                         EM3
001432  050512                         DH100
001434  052276                         DT100
001436  052406                         DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
001440  045512                         EM4
001442  050512                         DH100
001444  052276                         DT100
001446  052406                         DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
001450  045533                         EM5
001452  050512                         DH100
001454  052276                         DT100
001456  052436                         DF02

                                        ;ERROR 6
001460  045552                         EM6                     ;D TO C PARITY ERROR
001462  050512                         DH100
001464  052276                         DT100
001466  052472                         DF03

                                        ;ERROR 7
001470  045576                         EM7                     ;DRIVE DETECTED PARITY ERROR
001472  050512                         DH100
001474  052276                         DT100
```

```
1694  001476  052472                    DF03
1695
1696                                    ;ERROR 10
1697  001500  045632                    EM10                ;AC LOW
1698  001502  050512                    DH100
1699  001504  052276                    DT100
1700  001506  052436                    DF02
1701
1702                                    ;ERROR 11
1703  001510  045641                    EM11                ;SPEED LOSS
1704  001512  050512                    DH100
1705  001514  052276                    DT100
1706  001516  052436                    DF02
1707
1708                                    ;ERROR 12
1709  001520  045654                    EM12                ;ILLEGAL FUNCTION
1710  001522  050512                    DH100
1711  001524  052276                    DT100
1712  001526  052436                    DF02
1713
1714                                    ;ERROR 13
1715  001530  045675                    EM13                ;PROGRAMMING ERROR
1716  001532  050512                    DH100
1717  001534  052276                    DT100
1718  001536  052406                    DF01
1719
1720                                    ;ERROR 14           ;NON-EXISTANT FUNCTION
1721  001540  045717                    EM14
1722  001542  050512                    DH100
1723  001544  052276                    DT100
1724  001546  052436                    DF02
1725
1726                                    ;ERROR 15
1727  001550  045745                    EM15                ;DRIVE TYPE ERROR
1728  001552  050512                    DH100
1729  001554  052276                    DT100
1730  001556  052436                    DF02
1731
1732                                    ;ERROR 16
1733  001560  045766                    EM16                ;FORMAT ERROR
1734  001562  050512                    DH100
1735  001564  052276                    DT100
1736  001566  052436                    DF02
1737
1738                                    ;ERROR 17
1739  001570  046003                    EM17                ;WRITE LOCK ERROR
1740  001572  050512                    DH100
1741  001574  052276                    DT100
1742  001576  052436                    DF02
1743
1744                                    ;ERROR 20
1745  001600  046024                    EM20                ;DRIVE UNSAFE
1746  001602  050512                    DH100
1747  001604  052276                    DT100
1748  001606  052436                    DF02
1749
```

```
 1750                                   ;ERROR 21
 1751  001610  046041                   EM21                ;SEEK INCOMPLETE
 1752  001612  050512                   DH100
 1753  001614  052276                   DT100
 1754  001616  052436                   DF02
 1755
 1756                                   ;ERROR 22
 1757  001620  046061                   EM22                ;CYLINDER OVERFLOW
 1758  001622  050512                   DH100
 1759  001624  052276                   DT100
 1760  001626  052436                   DF02
 1761
 1762                                   ;ERROR 23
 1763  001630  046103                   EM23                ;ILLEGAL CYLINDER
 1764  001632  050512                   DH100
 1765  001634  052276                   DT100
 1766  001636  052436                   DF02
 1767
 1768                                   ;ERROR 24
 1769  001640  046134                   EM24                ;DRIVE OFF TRACK
 1770  001642  050512                   DH100
 1771  001644  052276                   DT100
 1772  001646  052436                   DF02
 1773
 1774                                   ;ERROR 25
 1775  001650  046154                   EM25                ;DRIVE TIMING ERROR
 1776  001652  050512                   DH100
 1777  001654  052276                   DT100
 1778  001656  052436                   DF02
 1779
 1780                                   ;ERROR 26
 1781  001660  046177                   EM26                ;DATA LATE
 1782  001662  050512                   DH100
 1783  001664  052276                   DT100
 1784  001666  052436                   DF02
 1785
 1786                                   ;ERROR 27
 1787  001670  046211                   EM27                ;CONTROLLER TIMEOUT
 1788  001672  050512                   DH100
 1789  001674  052276                   DT100
 1790  001676  052436                   DF02
 1791
 1792                                   ;ERROR 30           ;OPERATION INCOMPLETE
 1793  001700  046234                   EM30
 1794  001702  050512                   DH100
 1795  001704  052276                   DT100
 1796  001706  052546                   DF05
 1797
 1798                                   ;ERROR 31
 1799  001710  046261                   EM31                ;HEADER VRC ERROR
 1800  001712  050512                   DH100
 1801  001714  052276                   DT100
 1802  001716  052546                   DF05
 1803
 1804                                   ;ERROR 32
 1805  001720  046302                   EM32                ;DATA CHECK ERROR
```

```
1806  001722  050512                  DH100
1807  001724  052276                  DT100
1808  001726  052602                  DF07
1809
1810                                  ;ERROR 33
1811  001730  046323                  EM33          ;WRITE CHECK ERROR
1812  001732  050512                  DH100
1813  001734  052276                  DT100
1814  001736  052636                  DF10
1815
1816                                  ;ERROR 34
1817  001740  046345                  EM34          ;DATA MISCOMPARE(S)
1818  001742  050512                  DH100
1819  001744  052276                  DT100
1820  001746  052532                  DF04
1821
1822                                  ;ERROR 35
1823  001750  046365                  EM35          ;NO DRIVE RESPONSE-UFE & NXD
1824  001752  050512                  DH100
1825  001754  052276                  DT100
1826  001756  052406                  DF01
1827
1828                                  ;ERROR 36
1829  001760  046421                  EM36          ;DRIVE ERROR WILL NOT CLEAR
1830  001762  000000                  0
1831  001764  000000                  0
1832  001766  000000                  0
1833
1834                                  ;ERROR 37     ;DRIVE STATUS CHANGE WILL NOT CLEAR
1835  001770  046454                  EM37
1836  001772  000000                  0
1837  001774  000000                  0
1838  001776  000000                  0
1839
1840                                  ;ERROR 40
1841  002000  046517                  EM40          ;ATTENTION BUT NO STATUS CHANGE OR FAULT
1842  002002  050512                  DH100
1843  002004  052276                  DT100
1844  002006  052436                  DF02
1845
1846                                  ;ERROR 41
1847  002010  046563                  EM41          ;ATTENTION BUT DRIVE NOT AVAILABLE
1848  002012  050512                  DH100
1849  002014  052276                  DT100
1850  002016  052436                  DF02
1851
1852                                  ;ERROR 42
1853  002020  046621                  EM42          ;ATTENTION WHEN NOT EXPECTED
1854  002022  050512                  DH100
1855  002024  052276                  DT100
1856  002026  052436                  DF02
1857
1858                                  ;ERROR 43     ;ERROR WHILE GATHERING DRIVE STATUS
1859  002030  046651                  EM43
1860  002032  050512                  DH100
1861  002034  052276                  DT100
```

```
1862  002036  052702                        DF12
1863
1864                                        ;ERROR 44
1865  002040  047126                        EM63            ;CLEAR CONTROLLER DID NOT CLEAR ERROR
1866  002042  050512                        DH100
1867  002044  052276                        DT100
1868  002046  052702                        DF12
1869
1870                                        ;ERROR 45
1871  002050  047173                        EM64            ;NO ATTENTION IN ATTENTION SUMMARY REG
1872  002052  050512                        DH100
1873  002054  052276                        DT100
1874  002056  052702                        DF12
1875
1876                                        ;ERROR 46
1877  002060  047236                        EM65            ;UNSOLICITED ATTENTION
1878  002062  050512                        DH100
1879  002064  052276                        DT100
1880  002066  052702                        DF12
1881
1882                                        ;ERROR 47
1883  002070  047264                        EM66            ;UNEXPECTED DATA TYPE ERROR
1884  002072  050512                        DH100
1885  002074  052276                        DT100
1886  002076  052702                        DF12
1887
1888                                        ;ERROR 50
1889  002100  047317                        EM67            ;ATTENTION DID NOT RESET WITH CLEAR
1890  002102  050512                        DH100
1891  002104  052276                        DT100
1892  002106  052702                        DF12
1893
1894                                        ;ERROR 51
1895  002110  047356                        EM70            ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1896  002112  050512                        DH100
1897  002114  052276                        DT100
1898  002116  052702                        DF12
1899
1900                                        ;ERROR 52
1901  002120  046706                        EM52            ;MULTIPLE DRIVE SELECT
1902  002122  050512                        DH100
1903  002124  052276                        DT100
1904  002126  052702                        DF12
1905
1906                                        ;ERROR 53
1907  002130  046734                        EM53            ;ABREVIATED HCE ERROR
1908  002132  050512                        DH100
1909  002134  052276                        DT100
1910  002136  052732                        DF13
1911
1912                                        ;ERROR 54
1913  002140  046234                        EM30            ;OPERATION INCOMPLETE ERROR
1914  002142  050512                        DH100
1915  002144  052276                        DT100
1916  002146  052762                        DF14
1917
```

```
1918
1919  002150  046261                ;ERROR 55
1920  002152  050512                EM31                ;ABREVIATED HVRC ERROR
1921  002154  052276                DH100
1922  002156  052732                DT100
1923                                DF13
1924
1925  002160  046761                ;ERROR 56
1926  002162  050512                EM56                ;2 TIMEOUT ERROR
1927  002164  052276                DH100
1928  002166  053012                DT100
1929                                DF15
1930
1931  002170  046761                ;ERROR 57           ;2ND LEVEL IN SUBSYSTEM TIMEOUT
1932  002172  050512                EM56
1933  002174  052276                DH100
1934  002176  053056                DT100
1935                                DF16
1936
1937  002200  047000                ;ERROR 60
1938  002202  000000                EM60                ;ERROR IN RECAL FOR RECOVERY
1939  002204  000000                0
1940  002206  000000                0
1941                                0
1942
1943  002210  047034                ;ERROR 61
1944  002212  000000                EM61                ;ABORT MESSAGE
1945  002214  000000                0
1946  002216  000000                0
1947                                0
1948
1949  002220  047102                ;ERROR 62
1950  002222  050512                EM62                ;CYLINDER MISCOMPARE
1951  002224  052276                DH100
1952  002226  053122                DT100
1953                                DF17
1954
1955  002230  000000                ;ERROR 63           ;DATA ERROR WORDS
1956  002232  000000                0
1957  002234  052370                0
1958  002236  053232                DT602
1959                                DF25
1960
1961  002240  047126                ;ERROR 64
1962  002242  050512                EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
1963  002244  052276                DH100
1964  002246  052436                DT100
1965                                DF02
1966
1967  002250  047173                ;ERROR 65
1968  002252  050512                EM64                ;NO ATTENTION IN ATTENTION SUMMARY REG
1969  002254  052276                DH100
1970  002256  052436                DT100
1971                                DF02
1972
1973  002260  047236                ;ERROR 66
                                    EM65                ;UNSOLICITED ATTENTION
```

```
1974   002262   050512                  DH100
1975   002264   052276                  DT100
1976   002266   052436                  DF02
1977
1978                                    ;ERROR 67
1979   002270   047264                  EM66            ;UNEXPECTED DATA TYPE ERROR
1980   002272   050512                  DH100
1981   002274   052276                  DT100
1982   002276   052436                  DF02
1983
1984                                    ;ERROR 70
1985   002300   047317                  EM67            ;ATTENTION DID NOT RESET WITH CLEAR
1986   002302   050512                  DH100
1987   002304   052276                  DT100
1988   002306   052436                  DF02
1989
1990                                    ;ERROR 71
1991   002310   047356                  EM70            ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
1992   002312   050512                  DH100
1993   002314   052276                  DT100
1994   002316   052436                  DF02
1995
1996                                    ;ERROR 72
1997   002320   047425                  EM71            ;DATA LATE WHEN UNLOADING HEADER
1998   002322   050512                  DH100
1999   002324   052276                  DT100
2000   002326   052436                  DF02
2001
2002                                    ;ERROR 73
2003   002330   047465                  EM72            ;CONTROLLER ERROR DURING DRIVER SERVICE
2004   002332   050512                  DH100
2005   002334   052276                  DT100
2006   002336   052436                  DF02
2007
2008                                    ;ERROR 74
2009   002340   047534                  EM73            ;DRIVE DETECTED PARITY ERROR
2010   002342   050512                  DH100
2011   002344   052276                  DT100
2012   002346   052436                  DF02
2013
2014                                    ;ERROR 75
2015   002350   047570                  EM74            ;UNDEFINED ERROR
2016   002352   050512                  DH100
2017   002354   052276                  DT100
2018   002356   052436                  DF02
2019
2020                                    ;ERROR 76
2021   002360   047610                  EM75            ;MARKING SECTOR BAD MESSAGE
2022   002362   000000                  0
2023   002364   000000                  0
2024   002366   000000                  0
2025
2026                                    ;ERROR 77
2027   002370   047640                  EM76            ;BAD DATA VERIFICATION WITH READ
2028   002372   051477                  DH605
2029   002374   052362                  DT601
```

```
2030  002376  053156                    DF21
2031
2032                                    ;ERROR 100
2033  002400  047722                    EM77                ;RETRY SUCCESFUL MESSAGE
2034  002402  000000                    0
2035  002404  052362                    DT601
2036  002406  053216                    DF23
2037
2038                                    ;ERROR 101
2039  002410  047722                    EM77                ;ANOTHER RETRY SUCCESSFUL MESSAGE
2040  002412  000000                    0
2041  002414  052362                    DT601
2042  002416  053216                    DF23
2043
2044                                    ;ERROR 102
2045  002420  047743                    EM100               ;RETRY UNSUCCESSFUL MESSAGE
2046  002422  000000                    0
2047  002424  052362                    DT601
2048  002426  053216                    DF23
2049
2050                                    ;ERROR 103
2051  002430  047766                    EM101               ;NO VALID HEADERS IN TRACK JUST READ
2052  002432  051461                    DH6042
2053  002434  052362                    DT601
2054  002436  053226                    DF24
2055
2056                                    ;ERROR 104
2057  002440  050044                    EM102               ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2058  002442  050512                    DH100
2059  002444  052276                    DT100
2060  002446  052436                    DF02
2061
2062                                    ;ERROR  105
2063  002450  050116                    EM103               ;TIMED-OUT ON READ HEADER
2064  002452  050512                    DH100
2065  002454  052276                    DT100
2066  002456  052472                    DF03
2067
2068                                    ;ERROR  106
2069  002460  050144                    EM104               ;TIMED-OUT ON SEEK
2070  002462  050512                    DH100
2071  002464  052276                    DT100
2072  002466  052472                    DF03
2073
2074                                    ;ERROR  107
2075  002470  050166                    EM105               ;DRIVE SIEZED BY OTHER PORT
2076  002472  050512                    DH100
2077  002474  052276                    DT100
2078  002476  052436                    DF02
2079
2080                                    ;ERROR  110
2081  002500  050221                    EM106               ;"DATA MISCMPR WHILE BAI SET"
2082  002502  050512                    DH100
2083  002504  052276                    DT100
2084  002506  053246                    DF27
2085
```

```
2086
2087  002510  050254                ;ERROR  111
2088  002512  000000                EM107                   ;"NO NEM WHEN EXPECTED"
2089  002514  000000                0
2090  002516  000000                0
2091                                0
2092
2093  002520  050321                ;ERROR  112
2094  002522  050512                EM110                   ;"INTRPT WHEN CNTRLR NOT READY"
2095  002524  052276                DH100
2096  002526  052406                DT100
2097                                DF01
2098
2099  002530  050354                ;ERROR  113
2100  002532  050512                EM111                   ;"NO ATT'N ON SEEK"
2101  002534  052276                DH100
2102  002536  052436                DT100
2103                                DF02
2104
2105  002540  050375                ;ERROR  114
2106  002542  051706                EM112                   ;DRIVE'S CYLINDER INCORRECT
2107  002544  052336                DH702
2108  002546  053236                DT202
2109                                DF26
2110
2111  002550  000000                ;ERROR  115
2112  002552  000000                0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2113  002554  052362                0
2114  002556  052662                DT601
2115                                DF11
2116
2117  002560  046345                ;ERROR  116
2118  002562  051023                EM34                    ;DATA MISCOMPARE (11/70)
2119  002564  000000                DH103
2120  002566  053146                0
2121                                DF20
2122
2123  002570  000000                ;ERROR  117
2124  002572  000000                0                       ;PART OF DATA MISCOMPARE
2125  002574  052276                0
2126  002576  053166                DT100
2127                                DF22
2128
2129  002600  050430                ;ERROR  120
2130  002602  050512                EM113                   ;ABORT- CAN'T READ BSF
2131  002604  052276                DH100
2132  002606  052406                DT100
2133                                DF01
2134
2135  002610  050456                ;ERROR  121
2136  002612  050512                EM114                   ;KT11 FAILURE
2137  002614  052276                DH100
2138  002616  053272                DT100
2139                                DF30
2140
2141  002620  050473                ;ERROR  122
                                    EM115                   ;MEM PARITY ERROR
```

```
 2142  002622  050512                        DH100
 2143  002624  052276                        DT100
 2144  002626  053316                        DF31
 2145
 2146                                        ;ERROR  123
 2147  002630  045467                        EM3                 ;NED ON SIZING UNDER APT
 2148  002632  000000                        0
 2149  002634  000000                        0
 2150  002636  000000                        0
 2151
 2152
 2153
 2154
 2155
 2156
 2157
 2158
 2159
 2160
 2161
 2162
 2163
 2164
 2165
 2166
 2167
 2168
 2169
 2170
```

```
                        .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                        !
                        ;-----------------------------------------------------------
```

```
;RKDB (DATA BUFFER)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;DB15 ! DB14 ! DB13 ! DB12 ! DB11 ! DB10 ! DB9  ! DB8  !
;                    READ/WRITE                       !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
; DB7 ! DB6  ! DB5  ! DB4  ! DB3  ! DB2  ! DB1  ! DB0  !
;                    READ/WRITE                       !
;-----------------------------------------------------

;RKASOF (ATTENTION SUMMARY AND OFFSET)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;ATN7 ! ATN6 ! ATN5 ! ATN4 ! ATN3 ! ATN2 ! ATN1 ! ATN0 !
;                    READ ONLY                        !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
; UN  ! OF6  ! OF5  ! OF4  ! OF3  ! OF2  ! OF1  ! OF0  !
;  ?  !               READ/WRITE                       !
;-----------------------------------------------------

;RKWC (WORD COUNT)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;WC15 ! WC14 ! WC13 ! WC12 ! WC11 ! WC10 ! WC9  ! WC8  !
;                    READ/WRITE                       !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
; WC7 ! WC6  ! WC5  ! WC4  ! WC3  ! WC2  ! WC1  ! WC0  !
;                    READ/WRITE                       !
;-----------------------------------------------------

;RKBA (BUS ADDRESS)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;BA15 ! BA14 ! BA13 ! BA12 ! BA11 ! BA10 ! BA9  ! BA8  !
;                    READ/WRITE                       !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
; BA7 ! BA6  ! BA5  ! BA4  ! BA3  ! BA2  ! BA1  ! BA0  !
;                    READ/WRITE                !ALWYS0!
;-----------------------------------------------------
```

```
2276                    ;RKER (ERROR REGISTER)
2277                    ; 15     14     13     12     11     10     9      8
2278                    ;-----------------------------------------------------
2279                    ; DCK ! UNS  ! OPI  ! DTE  ! WLE  ! IDAE ! COE  ! HVRC !
2280                    ;                    READ ONLY                        !
2281                    ;-----------------------------------------------------
2282
2283                    ;  7      6      5      4      3      2      1      0
2284                    ;-----------------------------------------------------
2285                    ; BSE ! ECH  ! DTYPE! FMTE ! DRPAR! ILF  ! SKI  ! ILC  !
2286                    ;                    READ ONLY                        !
2287                    ;-----------------------------------------------------
2288
2289                    ;RKDS (DRIVE STATUS)
2290                    ; 15     14     13     12     11     10     9      8
2291                    ;-----------------------------------------------------
2292                    ;SVAL ! DSC  ! PIP  ! SPON ! WRL  !  UN  !  UN  ! DTP  !
2293                    ;                    READ ONLY                        !
2294                    ;-----------------------------------------------------
2295
2296                    ;  7      6      5      4      3      2      1      0
2297                    ;-----------------------------------------------------
2298                    ;DRDY !  VV  ! DROT ! SPLS ! ACLO ! OFSET!  UN  ! DRA  !
2299                    ;                    READ ONLY                        !
2300                    ;-----------------------------------------------------
2301
2302                    ;RKMR1 (MAINTENANCE REG 1)
2303                    ; 15     14     13     12     11     10     9      8
2304                    ;-----------------------------------------------------
2305                    ;RD    ! WRT  ! ECCW ! PCD  ! PCA  ! MEWD ! MERD ! MCLK !
2306                    ;GATE ! GATE !      !      !      !      !      !      !
2307                    ;             READ ONLY                 ! READ/WRITE  !
2308                    ;-----------------------------------------------------
2309
2310                    ;  7      6      5      4      3      2      1      0
2311                    ;-----------------------------------------------------
2312                    ;MIND ! MSP  ! DMD  ! PAT  ! MS3  ! MS2  ! MS1  ! MS0  !
2313                    ;                    READ/WRITE                       !
2314                    ;-----------------------------------------------------
```

```
                        ;RKECC/PAT
                        ; 15     14     13     12     11     10     9      8
                        ;-------------------------------------------------------
                        ; UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
                        ;                    READ ONLY                          !
                        ;-------------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-------------------------------------------------------
                        ;EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
                        ;                    READ ONLY                          !
                        ;-------------------------------------------------------
                        ;
                        ;RKECC/POS
                        ; 15     14     13     12     11     10     9      8
                        ;-------------------------------------------------------
                        ; UN  !  UN  !  UN  ! EPO12! EPO11! EPO10! EPO9 ! EPO8 !
                        ;                    READ ONLY                          !
                        ;-------------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-------------------------------------------------------
                        ;EPO7 ! EPO6 ! EPO5 ! EPO4 ! EPO3 ! EPO2 ! EPO1 ! EPO0 !
                        ;                    READ ONLY                          !
                        ;-------------------------------------------------------
                        ;
                        ;DRIVE STATUS INFORMATION
                        ;
                        ;LINE A MESSAGE 00
                        ; 15     14     13     12     11     10     9      8
                        ;-------------------------------------------------------
                        ; PAR ! DSC  ! PIP  ! SPON ! WRLK ! OFFON! FMT  ! DRTP !
                        ;-------------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-------------------------------------------------------
                        ;DRDY !  VV  ! DRA  !  NU  !  NU  ! DRIVE SELECT CODE  !
                        ;-------------------------------------------------------
                        ;
```

```
;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
```

```
2447            000012                  RKDS=    12                     ;DRIVE STATUS REGISTER
2448            000014                  RKER=    14                     ;ERROR REGISTER
2449            000016                  RKASOF=  16                     ;ATTENTION SUMMARY AND OFFSET REGISTER
2450            000020                  RKDC=    20                     ;DESIRED CYLINDER REGISTER
2451            000020                  RKDCYL=  20                     ;DESIRED CYLINDER REGISTER
2452            000024                  RKDB=    24                     ;DATA BUFFER
2453            000026                  RKMR1=   26                     ;MAINTENANCE REGISTER 1
2454            000034                  RKMR2=   34                     ;MAINTENANCE REGISTER 2
2455            000036                  RKMR3=   36                     ;MAINTENANCE REGISTER 3
2456            000030                  RKPOS=   30                     ;ECC POSITION INFORMATION
2457            000030                  RKECPS=  30                     ;ECC POSITION INFORMATION
2458            000032                  RKPAT=   32                     ;ECC PATTERN INFORMATION
2459            000032                  RKECPT=  32                     ;ECC PATTERN INFORMATION
2460
2461                                    .SBTTL   DRIVE COMMANDS
2462
2463            000101                  SELDRV=  101                    ;SELECT DRIVE
2464            000103                  PACK=    103                    ;PACK ACKNOWLEDGE
2465            000105                  CLEAR=   105                    ;DRIVE CLEAR
2466            000107                  UNLOAD=  107                    ;UNLOAD
2467            000111                  SRTSPL=  111                    ;START SPINDLE
2468            000113                  RECAL=   113                    ;RECALIBRATE
2469            000115                  OFFSET=  115                    ;OFFSET
2470            000117                  SEEK=    117                    ;SEEK
2471            000121                  RDDATA=  121                    ;READ DATA
2472            000123                  WRDATA=  123                    ;WRITE DATA
2473            000125                  RDHEAD=  125                    ;READ HEADER
2474            000127                  WRHEAD=  127                    ;WRITE HEADER AND DATA
2475            000131                  WRTCHK=  131                    ;WRITE CHECK
2476
2477                                    ;        THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
2478                                    ;        TO SIMULATE A SPECIFIC DESIRED OPERATION
2479
2480            000140                  RELEAS=  140                    ;RELEASE DRIVE
2481            000141                  RDSTAT=  141                    ;GET ALL STATUS FROM DRIVE
2482            000164                  RDALHD=  164                    ;READ ALL HEADERS
2483            000176                  CONCLR=  176                    ;CONTROLLER CLEAR (BIT 15 OF CS1)
2484            000177                  SUBCLR=  177                    ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
2485            000300                  INTR=    300                    ;GENERATE INTERRUPT TO CPU
2486
2487                                    ;        DRIVER ISSUED SERVICE COMMANDS
2488
2489            000001                  DR.SEL=  001                    ;DRIVE SELECT
2490            000005                  DR.CLR=  005                    ;DRIVE CLEAR
2491
2492                                    .SBTTL   CONTROL AND STATUS REGISTER 1 BITS
2493
2494            000001                  GO=      BIT0                   ;GO BIT
2495            000100                  IE=      BIT6                   ;INTERRUPT ENABLE
2496            000200                  RDY=     BIT7                   ;CONTROLLER READY
2497            000400                  BA16=    BIT8                   ;BUS ADDRESS BIT 16
2498            001000                  BA17=    BIT9                   ;BUS ADDRESS BIT 17
2499            002000                  CDT=     BIT10                  ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
2500            004000                  CTO=     BIT11                  ;CONTROLLER TIMED OUT WAITING FOR
2501                                                                    ;  DRIVE RESPONSE
2502            010000                  CFMT=    BIT12                  ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
```

```
                                ;CONTROL AND STATUS REGISTER 1 BITS

020000          SPAR=   BIT13           ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
040000          DI=     BIT14           ;DRIVE INTERRUPT
100000          CERR=   BIT15           ;CONTROLLER ERROR
100000          CCLR=   BIT15           ;CONTROLLER CLEAR

                ;       THESE BIT DEFINITIONS ARE USED FOR ADDRESS
                ;       THE HIGH BYTE OF RKCS1

000001          B.BA16= BIT0            ;BUS ADDRESS BIT 16
000002          B.BA17= BIT1            ;BUS ADDRESS BIT 17
000004          B.CDT=  BIT2            ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
000020          B.CFMT= BIT4            ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)

                .SBTTL  CONTROL AND STATUS REGISTER 2 BITS

000007          DRVMSK= 7               ;MASK FOR DRIVE SELECTION CODE
000010          DESL=   BIT3            ;DESELECT OR RELEASE DRIVE IN BITS 0-2
000010          RLS=    BIT3            ;DESELECT OR RELEASE DRIVE IN BITS 0-2
000020          BAI=    BIT4            ;BUS ADDRESS INCREMENT INHIBIT
000040          CLR=    BIT5            ;CLEAR CONTROLLER AND ALL DRIVES
000040          SCLR=   BIT5            ;CLEAR CONTROLLER AND ALL DRIVES
000100          IR=     BIT6            ;INPUT READY
000200          OR=     BIT7            ;OUTPUT READY
000400          UFE=    BIT8            ;UNIT FIELD ERROR
001000          MDS=    BIT9            ;MULTIPLE DRIVE SELECT
002000          PGE=    BIT10           ;PROGRAMMING ERROR
004000          NEM=    BIT11           ;NON-EXISTENT MEMORY
010000          NED=    BIT12           ;NON-EXISTENT DRIVE
020000          UPE=    BIT13           ;UNIBUS PARITY ERROR
040000          WCE=    BIT14           ;WRITE CHECK ERROR
100000          DLT=    BIT15           ;DATA LATE ERROR

                .SBTTL  ERROR REGISTER BIT DEFINITION

000001          ILC=    BIT0            ;ILLEGAL FUNCTION CODE
                .*ILF=  BIT0            ;ILLEGAL FUNCTION CODE
000002          SKI=    BIT1            ;SEEK INCOMPLETE
000004          ILF=    BIT2            ;ILLEGAL DRIVE FUNCTION
000004          NXF=    BIT2            ;ILLEGAL DRIVE FUNCTION
000010          DRPAR=  BIT3            ;DRIVE DETECTED DRIVE BUS PARITY ERROR
000020          FMTE=   BIT4            ;FORMAT ERROR
000040          DTYE=   BIT5            ;DRIVE TYPE ERROR
000100          ECH=    BIT6            ;ECC HARD
000200          BSE=    BIT7            ;BAD SECTOR ERROR
000400          HCRC=   BIT8            ;HEADER CRC ERRROR
000400          HVRC=   BIT8            ;HEADER VRC ERRROR
001000          COE=    BIT9            ;CYLINDER ADDRESS OVERFLOW ERROR
002000          IDAE=   BIT10           ;INVALID DISK ADDRESS ERROR
004000          WLE=    BIT11           ;WRITE LOCK ERROR
010000          DTE=    BIT12           ;DRIVE TIMING ERROR
020000          OPI=    BIT13           ;OPERATION (SEARCH) INCOMPLETE
040000          UNS=    BIT14           ;DRIVE UNSAFE
100000          DCK=    BIT15           ;DATA CHECK

                .SBTTL  STATUS REGISTER BIT DEFINITION
```

```
2559            000001                  DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
2560                                                                    ;  THIS BIT IS RESET)
2561            000004                  OFST=   BIT2                    ;DRIVE OFFSET
2562            000010                  ACLO=   BIT3                    ;AC LOW
2563            000020                  SPDLSS= BIT4                    ;SPEED LOSS
2564            000020                  DCLO=   BIT4                    ;DC LOW
2565            000040                  DROT=   BIT5                    ;DRIVE OFF TRACK
2566            000100                  VV=     BIT6                    ;VOLUME VALID
2567            000200                  DRY=    BIT7                    ;DRIVE READY
2568            000200                  DRDY=   BIT7                    ;DRIVE READY
2569            000400                  DDT=    BIT8                    ;DRIVE TYPE (0=RK06,1=RK07)
2570            004000                  WRL=    BIT11                   ;WRITE LOCK
2571            020000                  PIP=    BIT13                   ;POSITIONING IN PROGRESS
2572            040000                  DSC=    BIT14                   ;DRIVE STATUS CHANGE
2573            100000                  SVAL=   BIT15                   ;STATUS VALID
2574
2575                                    .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
2576
2577            000017                  MESMSK= 17                      ;MESSAGE MASK
2578
2579            000020                  PAT=    BIT4                    ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
2580            000040                  DMD=    BIT5                    ;DIAGNOSTIC MODE
2581            000100                  MSP=    BIT6                    ;MAINTENANCE SECTOR PULSE
2582            000200                  MIND=   BIT7                    ;MAINTENANCE INDEX
2583            000400                  MCLK=   BIT8                    ;MAINTENANCE CLOCK
2584            001000                  MERD=   BIT9                    ;MAINTENANCE ENCODED READ DATA
2585            002000                  MEWD=   BIT10                   ;MAINTENACNE ENCODED WRITE DATA
2586            004000                  PCA=    BIT11                   ;PRECOMPENSATION ADVANCE
2587            010000                  PCD=    BIT12                   ;PRECOMPENSATION DELAY
2588            020000                  ECCW=   BIT13                   ;ECC WORD IS BEING READ OR WRITTEN
2589            040000                  WRTGAT= BIT14                   ;WRITE GATE
2590            100000                  RDGATE= BIT15                   ;READ GATE
2591
2592                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
2593
2594            000040                  S.DRA=  BIT5                    ;DRIVE AVAILIABLE
2595            000100                  S.VV=   BIT6                    ;VOLUME VALID
2596            000200                  S.DRY=  BIT7                    ;DRIVE READY
2597            000400                  S.TYPE= BIT8                    ;DRIVE TYPE
2598            001000                  S.FORM= BIT9                    ;DRIVE FORMAT
2599            002000                  S.OFF=  BIT10                   ;OFFSET
2600            004000                  S.WRL=  BIT11                   ;WRITE LOCK
2601            010000                  S.SPIN= BIT12                   ;SPINDLE ON
2602            020000                  S.PIP=  BIT13                   ;POSITIONING IN PROGRESS
2603            040000                  S.DSC=  BIT14                   ;DRIVE STATUS CHANGE
2604
2605                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
2606
2607            000040                  S.ICYL= BIT5                    ;ILLEGAL CYLINDER ADDRESS
2608            000100                  S.ACLO= BIT6                    ;AC LOW
2609            000200                  S.FLT=  BIT7                    ;DRIVE FAULT
2610            000400                  S.ILF=  BIT8                    ;ILLEGAL FUNCTION
2611            001000                  S.PAR=  BIT9                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
2612            002000                  S.SKI=  BIT10                   ;SEEK INCOMPLETE
2613            004000                  S.WLE=  BIT11                   ;WRITE LOCK ERROR
2614            010000                  S.SPLS= BIT12                   ;SPEED LOSS
```

```
                                DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

2615            010000          S.DCLO= BIT12                   ;DC LOW
2616            020000          S.DROT= BIT13                   ;DRIVE OFF TRACK
2617            040000          S.UNS=  BIT14                   ;DRIVE UNSAFE
2618
2619                            .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
2620
2621            000020          S.XDOK= BIT4                    ;TRANSDUCER OK
2622            000040          S.HDHM= BIT5                    ;HEADS HOME
2623            000100          S.BRHM= BIT6                    ;BRUSHES HOME
2624            000200          S.DOOR= BIT7                    ;DOOR INTERLOCKED
2625            000400          S.CART= BIT8                    ;CARTRAGE INTERLOCK
2626            001000          S.SPOK= BIT9                    ;SPEED OK
2627            002000          S.FWD=  BIT10                   ;FORWARD
2628            004000          S.REV=  BIT11                   ;REVERSE
2629            010000          S.LOAD= BIT12                   ;HEADS LOADING
2630            020000          S.RTZ=  BIT13                   ;RETURN TO ZERO
2631            040000          S.UNLD= BIT14                   ;HEADS UNLOADING
2632
2633                            .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
2634
2635            000020          S.SECT= BIT4                    ;SECTOR ERROR
2636            000040          S.WCLK= BIT5                    ;WRITE CLOCK AND NO WRITE GATE
2637            000100          S.WGAT= BIT6                    ;WRITE GATE AND NO TRANSISTIONS
2638            000200          S.HDFL= BIT7                    ;HEAD FAULT
2639            000400          S.MHD=  BIT8                    ;MULTIPLE HEAD SELECT
2640            001000          S.XERR= BIT9                    ;INDEX ERROR
2641            002000          S.DIB=  BIT10                   ;DIBIT ERROR
2642            004000          S.PLO=  BIT11                   ;PLO ERROR
2643            010000          S.NMOV= BIT12                   ;SEEK AND NO MOTION
2644            020000          S.LIMD= BIT13                   ;LIMIT DETECT ON SEEK
2645            040000          S.BRKE= BIT14                   ;SERVO-BRAKE
2646
2647                            .SBTTL  COMMON MASKS
2648
2649    000007                  M.DRV=  7                       ;DRIVE CODE
2650    100000                  M.PAR=  BIT15                   ;PARITY
2651    000003                  M.ID=   3                       ;BYTE ID
2652    017760                  M.CDIF= 17760                   ;CYLINDER DIFFERENCE/OFFSET
2653    017760                  M.CADD= 17760                   ;CYLINDER ADDRESS
2654    077770                  M.SER=  77770                   ;DRIVE SERIAL NUMBER
2655    000760                  M.SECT= 760                     ;SECTOR COUNT
2656    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/RK07 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

002640      000     PARM0:  .BYTE   0                               ;DRIVE NUMBER
002641      000             .BYTE   0                               ;COMMAND
002642   000000             .WORD   0                               ;CYLINDER ADDRESS
002644      000             .BYTE   0                               ;SECTOR ADDRESS
002645      000             .BYTE   0                               ;TRACK ADDRESS
002646      000             .BYTE   0                               ;OFFSET VALUE
002647      000             .BYTE   0                               ;BUS ADDRESS (BITS 16 AND 17)
002650   000000             .WORD   0                               ;BUS ADDRESS (BITS 0 - 15)
002652   000000             .WORD   0                               ;WORD COUNT (2'S COMPLEMENT)
002654   000000             .WORD   0                               ;PROGRAM DRIVE STATUS INFORMATION
002656   000000             .WORD   0                               ;COMMAND AND STATUS REGISTER 1
002660   000000             .WORD   0                               ;COMMAND AND STATUS REGISTER 2
002662   000000             .WORD   0                               ;WORD COUNT REGISTER
002664   000000             .WORD   0                               ;BUS ADDRESS REGISTER
002666   000000             .WORD   0                               ;DESIRED TRACK AND SECTOR REGISTER
```

```
2769  002670  000000                    .WORD   0               ;DESIRED CYLINDER REGISTER
2770  002672  000000                    .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
2771  002674  000000                    .WORD   0               ;ERROR REGISTER
2772  002676  000000                    .WORD   0               ;STATUS REGISTER
2773  002700  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
2774  002702  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
2775  002704  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
2776  002706  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
2777  002710  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
2778  002712  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
2779  002714  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
2780  002716  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
2781  002720  000000                    .WORD   0               ;ECC POSITION INFORMATION
2782  002722  000000                    .WORD   0               ;ECC PATTERN INFORMATION
2783
2784                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
2785
2786  002724     000          PARM1:  .BYTE   0               ;DRIVE NUMBER
2787  002725     000                  .BYTE   0               ;COMMAND
2788  002726  000000                  .WORD   0               ;CYLINDER ADDRESS
2789  002730     000                  .BYTE   0               ;SECTOR ADDRESS
2790  002731     000                  .BYTE   0               ;TRACK ADDRESS
2791  002732     000                  .BYTE   0               ;OFFSET VALUE
2792  002733     000                  .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
2793  002734  000000                  .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
2794  002736  000000                  .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
2795  002740  000000                  .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
2796  002742  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 1
2797  002744  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 2
2798  002746  000000                  .WORD   0               ;WORD COUNT REGISTER
2799  002750  000000                  .WORD   0               ;BUS ADDRESS REGISTER
2800  002752  000000                  .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
2801  002754  000000                  .WORD   0               ;DESIRED CYLINDER REGISTER
2802  002756  000000                  .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
2803  002760  000000                  .WORD   0               ;ERROR REGISTER
2804  002762  000000                  .WORD   0               ;STATUS REGISTER
2805  002764  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
2806  002766  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
2807  002770  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
2808  002772  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
2809  002774  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
2810  002776  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
2811  003000  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
2812  003002  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
2813  003004  000000                  .WORD   0               ;ECC POSITION INFORMATION
2814  003006  000000                  .WORD   0               ;ECC PATTERN INFORMATION
2815
2816                          .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
2817
2818  003010  000000          T.CS1:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
2819                                                          ;  REGISTER 1
2820  003012  000000          T.CS2:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
2821                                                          ;  REGISTER 2
2822  003014  000000          T.WCR:  .WORD   0               ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
2823  003016  000000          T.BA:   .WORD   0               ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
2824  003020  000000          T.DA:   .WORD   0               ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
2825   003022   000000                  T.DC:   .WORD   0          ;TEMPORARY STORAGE FOR DRIVE CYLINDER
2826   003024   000000                  T.ASOF: .WORD   0          ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
2827                                                               ;  AND OFFSET
2828   003026   000000                  T.ER:   .WORD   0          ;TEMPORARY STORAGE FOR ERROR REGISTER
2829   003030   000000                  T.DS:   .WORD   0          ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
2830   003032   000000                  T.MR1:  .WORD   0          ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
2831   003034   000000                  T.MR2:  .WORD   0          ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
2832   003036   000000                  T.MR3:  .WORD   0          ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
2833   003040   000000                  T.POS:  .WORD   0          ;TEMPORARY STORAGE FOR ECC POSITION
2834   003042   000000                  T.PAT:  .WORD   0          ;TEMPORARY STORAGE FOR ECC PATTERN
2835   003044   000000                  T.DB:   .WORD   0          ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
2836
2837                                   .SBTTL  DRIVER PARAMERTERS
2838
2839   003046   177440                  RKBAS:  .WORD   177440     ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
2840   003050   000210                  RKVEC:  .WORD   210        ;ADDRESS OF R611 VECTOR
2841   003052   000240                  RKPRI:  .WORD   PR5        ;RK611 INTERRUPT PRIORITY
2842   003054   030272                  A.NORM: ERRFRE             ;ADDRESS OF NORMAL RETURN FROM DRIVER
2843   003056   031504                  A.ABNL: ERRHDL             ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
2844   003060   031000                  A.CONT: CONERR             ;ADDRESS OF CONTROLLER ERROR RETURN
2845   003062   000000                  E.CONT: .WORD   0          ;CONTROLLER ERROR STATUS
2846                                                               ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
2847                                                               ;  IS INITIATED.  IF A CONTROLLER ERROR
2848                                                               ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
2849                                                               ;  USED:
2850                                                               ;
2851            000001                 E.CCLR= BIT0               ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2852            000002                 E.NOAT= BIT1               ;NO ATTENTION IN ATTENTION SUMMARY REG
2853            000004                 E.UATT= BIT2               ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
2854            000010                 E.UDAT= BIT3               ;UNEXPECTED DATA TYPE ERROR
2855            000020                 E.CLAT= BIT4               ;ATTENTION DID NOT RESET WITH CLEAR
2856            000040                 E.SCLR= BIT5               ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
2857                                                               ;  ATTENTION
2858            000100                 E.ILLD= BIT6               ;ILLEGAL DRIVER COMMAND
2859            000400                 E.DLT=  BIT8               ;DATA LATE WHEN UNLOADING HEADER
2860            001000                 E.CERR= BIT9               ;CONTROLLER ERROR DURING DRIVER SERVICING
2861            002000                 E.DPAR= BIT10              ;DRIVE DETECTED PARITY ERROR
2862            040000                 E.CMTO= BIT14              ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
2863            100000                 E.MDS=  BIT15              ;MULTIPLE DRIVE SELECT
2864
2865   003064   000000                  O.WAIT: .WORD   0          ;PARAMETER BLOCK OF THE DRIVE
2866                                                               ;WAITING FOR COMMAND COMPLETION
2867   003066   000400                  W.MTIM: .WORD   400        ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
2868   003070   000400                  W.MILI: .WORD   400        ;16 MILISECOND TIME FOR PROGRAM
2869                                                               ;
2870                                                               ;       CPU     VALUE
2871                                                               ;       ---     -----
2872                                                               ;
2873                                                               ;       11/05   100
2874                                                               ;       11/10
2875                                                               ;       11/20
2876                                                               ;       11/34
2877                                                               ;       11/40
2878                                                               ;       11/45   400
2879                                                               ;       11/50
2880                                                               ;       11/70
```

```
2881  003072  000300                  W.SEC:   .WORD   300            ;SECOND COUNT COUNT FOR ALL COMMANDS
2882                                                                  ;  EXCEPT START SPINDLE
2883  003074  003000                  W.8SEC:  .WORD   3000           ;8 SECOND FOR DRIVE CYCLE DOWN
2884  003076  030000                  W.MIN:   .WORD   30000          ;MINUTE TIME FOR START SPINDLE
2885  003100  000000                  HDR.AD:  .WORD   0              ;ADDRESS USED FOR READ ALL HEADERS
2886  003102  000000                  HDR.CT:  .WORD   0              ;NUMBER OF HEADERS LEFT TO READ FOR READ
2887                                                                  ;  ALL HEADERS
2888  003104      000                 I.ISRL:  .BYTE   0              ;INTERRUPT OR RELEASED COMMAND ISSUED
2889  003105      002     004   010   H.HEAD:  .BYTE   2,4,10         ;HEAD DECODES
2890  003110      000                 W.TIME:  .BYTE   0              ;DRIVES BEING WATCH-DOG TIMED
2891
2892                                  .SBTTL   INTERRUPT MASKS
2893
2894  003111      000                 INTMSK:  .BYTE   0              ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
2895
2896                                  ;        INTERRUPT MASK TABLE
2897
2898  003112      001                 I.DRV:   .BYTE   1              ;INTERRUPT MASK FOR DRIVE 0
2899  003113      002                          .BYTE   2              ;INTERRUPT MASK FOR DRIVE 1
2900  003114      004                          .BYTE   4              ;INTERRUPT MASK FOR DRIVE 2
2901  003115      010                          .BYTE   10             ;INTERRUPT MASK FOR DRIVE 3
2902  003116      020                          .BYTE   20             ;INTERRUPT MASK FOR DRIVE 4
2903  003117      040                          .BYTE   40             ;INTERRUPT MASK FOR DRIVE 5
2904  003120      100                          .BYTE   100            ;INTERRUPT MASK FOR DRIVE 6
2905  003121      200                          .BYTE   200            ;INTERRUPT MASK FOR DRIVE 7
2906
2907                                  .SBTTL   PARAMETER BLOCK TABLE
2908
2909  003122  002640                  PBLKT:   PARM0                  ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
2910                                                                  ;DRIVE CALL.  MUST BE LOADED INTO PBLKT
2911
2912
2913                                  .SBTTL   TIME FOR WATCH-DOG TIMER
2914
2915  003124  000000                  W.DRV:   .WORD   0              ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
2916                                  .SBTTL   PROGRAM SPECIFIC RESERVED LOCATIONS
2917  003126      000                 MDFLAG:  .BYTE   0              ;FLAG TO INDIC. DEFLT OR PARAM MODE
2918  003127      000                 TSTING:  .BYTE   0              ;CURRENTLY RUNNING TESTS IF = 1
2919  003130      000                 DERCNT:  .BYTE   0              ;DATA ERROR COUNT
2920  003131      000                 OPCOMP:  .BYTE   0              ;OPERATION COMPLETE FLAG
2921  003132      000                 DONE:    .BYTE   0              ;DONE SWITCH
2922  003133      000                 TYPFMT:  .BYTE   0              ;DRIVE TYPE & FORMAT CONTROL
2923  003134      000                 FORMAT:  .BYTE   0              ;DRIVE FORMAT IN BIT 4 OF BYTE
2924  003135      000                 ERRCNT:  .BYTE   0              ;ERROR COUNT
2925  003136      004                 ERRLMT:  .BYTE   4              ;ERROR LIMIT
2926  003137      000                 DRVERS:  .BYTE   0              ;ERROR COUNT FOR CURRENT DRIVE
2927  003140      000                 OPCONT:  .BYTE   0              ;OPERATION CONTROL SWITCHES
2928  003141      000                 DRNAFG:  .BYTE   0              ;=1 INDICATES DRIVE SIEZED BY OTHER PORT
2929  003142      000                 REISSU:  .BYTE   0              ;DUAL-ACC FLAG TO RE-ISSUE COMMAND
2930  003143      000                 TSTTYP:  .BYTE   0              ;OFFSET TEST IDENTIFIER
2931  003144      000                 DCEFLG:  .BYTE   0              ;DATA CHECK ERROR FLAG
2932  003145      000                 WCEFLG:  .BYTE   0              ;WRITE CHECK ERROR FLAG
2933  003146      000                 DLTFLG:  .BYTE   0              ;DATA LATE ERROR FLAG
2934  003147      000                 DIF100:  .BYTE   0              ;"CYL DIF = 100/200" FLAG
2935  003150      000                 NORTRY:  .BYTE   0              ;"NO-RETRY" FLAG
2936  003151      000                 MEMABT:  .BYTE   0              ;SET BYTE = 1 FOR NO ABORT
```

```
2937                                                    ;     ON MEMORY PARITY ERRORS
2938  003152    000                 HLPOVL: .BYTE   0   ;HELP FILE OVERLAID INDICATOR
2939  003153    000                 NOTYPE: .BYTE   0   ;INDICATES PROGRAM JUST LOADED IF 0
2940  003154    000                 LOGDRV: .BYTE   0   ;LOGICAL DRIVE NUMBER (00-17)
2941  003155    000                 PASSNO: .BYTE   0   ;PASS NUMBER (= 1 OR 2)
2942  003156    000                 OFSDIR: .BYTE   0   ;OFFSET DIRECTION FLAG (0=NEG)
2943  003157    000                 OFSCNT: .BYTE   0   ;OFFSET COUNT FOR ONE DIRECTION
2944          000001                WHDSW=BIT0          ;WRITE HEADER & DATA SWITCH
2945          000002                VFHDSW=BIT1         ;VERIFY HEADERS SWITCH
2946          000004                WCDASW=BIT2         ;WRITE CHECK DATA SWITCH
2947          000010                RCDASW=BIT3         ;READ CHECK DATA SWITCH
2948          000020                OREQSW=BIT4         ;OFFSET REQUIRED SWITCH
2949                                      .EVEN
2950
2951          000114                MEMVEC=114          ;MEMORY PARITY TRAP VECTOR
2952  003160  000000                SAVWRD: .WORD   0   ;SCRATCH WORD
2953  003162  000400                BSSOFT: .BLKW   ↑D256 ;RECORD OF BAD SECTORS FROM SOFTWARE
2954  004162  000400                BSFACT: .BLKW   ↑D256 ;RECORD OF BAD SECTORS FROM FACTORY
2955  005162  000000  000000  000000 PRVCMD: .WORD   0,0,0,0,0,0 ;PREVIOUS COMMAND STORAGE
2956  005170  000000  000000  000000
2957  005176  000000  000000  000000 COMSTR: .WORD   0,0,0,0,0,0 ;CURRENT COMMAND STORAGE
2958  005204  000000  000000  000000
2959  005212  000102                BUFFO:  .BLKW   ↑D66 ;OUTPUT BUFFER 1
2960  005416  000000                LOWOCT: .WORD   0   ;LOW 16 BITS OF CONVERTED BINARY NUMBER
2961  005420  000000                HIGOCT: .WORD   0   ;HIGH BITS OF CONVERTED BINARY NO.
2962  005422  000000                BUFPRT: .WORD   0   ;BUFFER POINTER
2963  005424  000000                RECODE: .WORD   0   ;RECOVERY CODE WORD
2964  005426  000000                ERRCOM: .WORD   0   ;ERROR COMMAND
2965  005430  000000                DRIVE:  .WORD   0   ;NO.  OF  DRIVE IN USE
2966  005432  000000                STALLS: .WORD   0   ;CURRENT NO. OF UNIT STALLS TO APPLY
2967  005434  000000                CYLNDR: .WORD   0   ;CURRENT CYLINDER NUMBER
2968  005436  000000                OFINUS: .WORD   0   ;OFFSET IN USE
2969  005440  000000                TRACK:  .WORD   0   ;TRACK IN USE
2970  005442  000000                INTCHR: .WORD   0   ;TTY INTERRUPT INPUT WORD
2971  005444  000000                SCRACH: .WORD   0   ;ALL-PURPOSE SCRATCH WORD
2972  005446  000000                PATRN:  .WORD   0   ;DATA PATTERN LIST WORD
2973  005450  000000                PLOFST: .WORD   0   ;(+) OFFSET VALUE
2974  005452  000000                NGOFST: .WORD   0   ;(-) OFFSET VALUE
2975  005454  000005                SAVPRS: .BLKW   5   ;SAVE INITIAL PARAMS FOR XFER
2976  005466  000000                WDSXFR: .WORD   0   ;NO. OF WORDS ACTUALLY XFERRED
2977  005470  000000                FINCYL: .WORD   0   ;FINAL CYLINDER ADRS
2978  005472    000                 FINTRK: .BYTE   0   ;FINAL TRACK ADRS
2979  005473    000                 FINSEC: .BYTE   0   ;FINAL SECTOR ADRS
2980  005474  000000                LASTWC: .WORD   0   ;ACTUAL FINAL WC
2981  005476  000000                NEWON:  .WORD   0   ;NEW LOOK AT ONLINE DRIVES
2982  005500  000002                MA:     .BLKW   2
2983  005504  000002                PMA:    .BLKW   2   ;MEM. BUFFER ADDRESS
2984  005510  000000                SUBSYS: .WORD   0   ;CURRENT SUBSYSTEM NAME STORAGE (ASCII)
2985  005512  000021                DRVLST: .BLKW   17. ;EACH WORD CONTAINS THE NAME OF A DRIVE
2986                                                    ;  TO BE TESTED. HIGH BYTE IS SUBSYSTEM
2987                                                    ;  NAME, LOW BYTE IS DRIVE NO. (ASCII).
2988                                                    ;  IF WORD = 0, DRIVE IS NOT TESTED.
2989  005554  000000                DRVPTR: .WORD   0   ;DRIVE LIST POINTER
2990  005556  000010                SCRLST: .BLKB   8.  ;SCRATCH DRIVE LIST
2991
2992  005566                        PATOO:
```

```
2993
2994
2995
2996
2997                                    ;TABLE A  -   BASIC READ/WRITE TEST SECTORS FOR RK06
2998  005566                            TABLEA:
2999  005566  000620  000000                .WORD   620,0           ;DRIVE  0
3000  005572  000620  000002                .WORD   620,2           ;       1
3001  005576  000620  000004                .WORD   620,4           ;       2
3002  005602  000620  000006                .WORD   620,6           ;       3
3003  005606  000620  000010                .WORD   620,10          ;       4
3004  005612  000620  000012                .WORD   620,12          ;       5
3005  005616  000620  000014                .WORD   620,14          ;       6
3006  005622  000620  000016                .WORD   620,16          ;       7
3007  005626  000622  000000                .WORD   622,0           ;       10
3008  005632  000622  000002                .WORD   622,2           ;       11
3009  005636  000622  000004                .WORD   622,4           ;       12
3010  005642  000622  000006                .WORD   622,6           ;       13
3011  005646  000622  000010                .WORD   622,10          ;       14
3012  005652  000622  000012                .WORD   622,12          ;       15
3013  005656  000622  000014                .WORD   622,14          ;       16
3014  005662  000622  000016                .WORD   622,16          ;       17
3015
3016                                    ;TABLE A -  CORRESPONDING CYL NUMBERS FOR THE RK07
3017
3018  005666  001444  000000            TABL7A: .WORD   1444,0          ;DRV 0
3019  005672  001444  000002                    .WORD   1444,2          ;1
3020  005676  001444  000004                    .WORD   1444,4          ;2
3021  005702  001444  000006                    .WORD   1444,6          ;3
3022  005706  001444  000010                    .WORD   1444,10         ;4
3023  005712  001444  000012                    .WORD   1444,12         ;5
3024  005716  001444  000014                    .WORD   1444,14         ;6
3025  005722  001444  000016                    .WORD   1444,16         ;7
3026  005726  001446  000000                    .WORD   1446,0          ;10
3027  005732  001446  000002                    .WORD   1446,2          ;11
3028  005736  001446  000004                    .WORD   1446,4          ;12
3029  005742  001446  000006                    .WORD   1446,6          ;13
3030  005746  001446  000010                    .WORD   1446,10         ;14
3031  005752  001446  000012                    .WORD   1446,12         ;15
3032  005756  001446  000014                    .WORD   1446,14         ;16
3033  005762  001446  000016                    .WORD   1446,16         ;17
3034
3035
3036
3037                                    ;TABLE B  -   WORST CASE DATA PATTERN
3038  005766                            TABLEB:
3039  005766  072307                           072307
3040  005770  135143                           135143
3041  005772  156461                           156461
3042  005774  167230                           167230
3043  005776  073514                           073514
3044  006000  035646                           035646
3045  006002  016723                           016723
3046  006004  107351                           107351
3047  006006  143564                           143564
3048  006010  061672                           061672
```

```
3049  006012  030735                    030735
3050  006014  114356                    114356
3051  006016  046167                    046167
3052  006020  123073                    123073
3053  006022  151453                    151453
3054  006024  164616                    164616
3055
3056
3057
3058                              ;TABLE OF STARTING CYLINDERS FOR ALL OVWRT AND COMPAT CYL BLKS FOR RK06
3059  006026  000000  000060      TABLEC: .WORD   0,60            ;FOR CURRENT ZONE 1
3060  006032  000100  000160              .WORD   100,160         ;                 2
3061  006036  000200  000260              .WORD   200,260         ;                 3
3062  006042  000300  000360              .WORD   300,360         ;                 4
3063  006046  000400  000460              .WORD   400,460         ;                 5
3064  006052  000500  000560              .WORD   500,560         ;                 6
3065  006056  000600                      .WORD   600             ;                 7
3066
3067                              ;TABLE C  CORRESPONDING CYL NUMBERS FOR RK07
3068
3069  006060  000000  000160      TABL7C: .WORD   0,160           ;CURRENT ZONE 1
3070  006064  000200  000360              .WORD   200,360         ;2
3071  006070  000400  000560              .WORD   400,560         ;3
3072  006074  000600  000760              .WORD   600,760         ;4
3073  006100  001000  001160              .WORD   1000,1160       ;5
3074  006104  001200  001360              .WORD   1200,1360       ;6
3075  006110  001400                      .WORD   1400            ;7
3076
3077
3078
3079                              ;CYLINDER BLOCK LAYOUT TABLE
3080  006112  000020              TABLED: .BLKB   16.             ;CYLINDER 0 IN BLK
3081  006132  000020                      .BLKB   16.             ;          1
3082  006152  000020                      .BLKB   16.             ;          2
3083  006172  000020                      .BLKB   16.             ;          3
3084  006212  000020                      .BLKB   16.             ;          4
3085  006232  000020                      .BLKB   16.             ;          5
3086  006252  000020                      .BLKB   16.             ;          6
3087  006272  000020                      .BLKB   16.             ;          7
3088  006312  000020                      .BLKB   16.             ;          10
3089  006332  000020                      .BLKB   16.             ;          11
3090  006352  000020                      .BLKB   16.             ;          12
3091  006372  000020                      .BLKB   16.             ;          13
3092  006412  000020                      .BLKB   16.             ;          14
3093  006432  000020                      .BLKB   16.             ;          15
3094  006452  000020                      .BLKB   16.             ;          16
3095  006472  000020                      .BLKB   16.             ;          17
3096
3097
3098
3099                              ;TABLE G - PSEUDO-RANDOM DATA PATTERN
3100  006512                      TABLEG:
3101  006512  040135                      040135
3102  006514  177070                      177070
3103  006516  070414                      070414
3104  006520  064531                      064531
```

```
3105  006522  174473                          174473
3106  006524  062422                          062422
3107  006526  114352                          114352
3108  006530  036620                          036620
3109  006532  010031                          010031
3110  006534  052336                          052336
3111  006536  017310                          017310
3112  006540  011347                          011347
3113  006542  102367                          102367
3114  006544  152567                          152567
3115  006546  001246                          001246
3116  006550  160073                          160073
3117
3118
3119
3120                                  ;LIST OF STARTING DRIVE NUMBERS, TO GENERATE TABLED
3121  006552      000     001     003 STDLST: .BYTE   0,1,3,6,12,17,5,14,4,15,7,2,16,13,11,10
3122  006555      006     012     017
3123  006560      005     014     004
3124  006563      015     007     002
3125  006566      016     013     011
3126  006571      010
3127
3128
3129
3130                                  ;LIST OF USEABLE SECTORS IN EACH CYLINDER BLOCK
3131  006572      000     001     002 SECLST: .BYTE   0,1,2,3,5,6,7,10,12,13,14,15,17,20,21,22
3132  006575      003     005     006
3133  006600      007     010     012
3134  006603      013     014     015
3135  006606      017     020     021
3136  006611      022
3137
3138
3139
3140                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR NEG OFST
3141  006612  000020                  NOSCR0: .BLKW   16.
3142
3143                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR NEG OFST
3144  006652  000020                  NOSCR1: .BLKW   16.
3145
3146                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR NEG OFST
3147  006712  000020                  NOSCR2: .BLKW   16.
3148
3149
3150
3151                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 0 FOR POS OFST
3152  006752  000020                  POSCR0: .BLKW   16.
3153
3154                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 1 FOR POS OFST
3155  007012  000020                  POSCR1: .BLKW   16.
3156
3157                                  ;TABLE OF OVERWRITE TEST SCORES FOR TRACK 2 FOR POS OFST
3158  007052  000020                  POSCR2: .BLKW   16.
3159
3160
```

```
3161
3162
3163  007112  000020                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR NEG OFST
                                    NCSCR0: .BLKW   16.
3164
3165                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR NEG OFST
3166  007152  000020                NCSCR1: .BLKW   16.
3167
3168                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR NEG OFST
3169  007212  000020                NCSCR2: .BLKW   16.
3170
3171
3172
3173                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 0 FOR POS OFST
3174  007252  000020                PCSCR0: .BLKW   16.
3175
3176                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 1 FOR POS OFST
3177  007312  000020                PCSCR1: .BLKW   16.
3178
3179                                ;TABLE OF DATA COMPATIBILITY TEST SCORES FOR TRACK 2 FOR POS OFST
3180  007352  000020                PCSCR2: .BLKW   16.
3181
3182
3183
3184
3185                                ;SELF-TEST SCORE FOR TRACK 0 FOR NEG OFST
3186  007412  000000                NSSCR0: .WORD   0
3187
3188                                ;SELF-TEST SCORE FOR TRACK 1 FOR NEG OFST
3189  007414  000000                NSSCR1: .WORD   0
3190
3191                                ;SELF-TEST SCORE FOR TRACK 2 FOR NEG OFST
3192  007416  000000                NSSCR2: .WORD   0
3193
3194
3195
3196                                ;SELF-TEST SCORE FOR TRACK 0 FOR POS OFST
3197  007420  000000                PSSCR0: .WORD   0
3198
3199                                ;SELF-TEST SCORE FOR TRACK 1 FOR POS OFST
3200  007422  000000                PSSCR1: .WORD   0
3201
3202                                ;SELF-TEST SCORE FOR TRACK 2 FOR POS OFST
3203  007424  000000                PSSCR2: .WORD   0
3204
3205
3206
3207                                ;AVERAGE OF OVRWRT SCORES OVER ALL SURFACES
3208  007426  000000                OVRAVG: .WORD   0
3209
3210                                ;AVERAGE OF READ SCORES OVER ALL SURFACES
3211  007430  000000                COMAVG: .WORD   0
3212
3213
3214
3215                                ;SCRATCH TABLE OF SCORES FOR TRACK 0 FOR NEG OFST
3216  007432  000020                NSCOR0: .BLKB   16.
```

```
                                ;SCRATCH TABLE OF SCORES FOR TRACK 1 FOR NEG OFST
007452  000020                  NSCOR1: .BLKB   16.

                                ;SCRATCH TABLE OF SCORES FOR TRACK 2 FOR NEG OFST
007472  000020                  NSCOR2: .BLKB   16.



                                ;SCRATCH TABLE OF SCORES FOR TRACK 0 FOR POS OFST
007512  000020                  PSCOR0: .BLKB   16.

                                ;SCRATCH TABLE OF SCORES FOR TRACK 1 FOR POS OFST
007532  000020                  PSCOR1: .BLKB   16.

                                ;SCRATCH TABLE OF SCORES FOR TRACK 2 FOR POS OFST
007552  000020                  PSCOR2: .BLKB   16.



        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
        000400                  LEV2ER=BIT8                     ;LEVEL TWO ERROR
        001000                  BADSEC=BIT9                     ;BAD SECTOR FLAG
        002000                  TWOTOS=BIT10                    ;TWO TIME OUTS
        004000                  RCLREQ=BIT11                    ;RECALIBRATE REQUIRED
        010000                  DRNAVL=BIT12                    ;DRIVE NOT RELSD BY OTHER PORT
        100000                  ANYDER=BIT15                    ;ANY ERROR DETECTED FLAG

007572  005015  041412  051132  DZR6Q:  .ASCIZ  <15><12><12>/CZR6QB0 - RK06-RK07 DRIVE COMPATIBILITY PROGRAM/<15><12><12
007600  050466  030102  026440
007606  051040  030113  026466
007614  045522  033460  042040
007622  044522  042526  041440
007630  046517  040520  044524
007636  044502  044514  054524
007644  050040  047522  051107
007652  046501  005015  000012
007660  051012  030113  026466  RKBADR: .ASCIZ  <12>/RK06-07 BUS ADR = /
007666  033460  041040  051525
007674  040440  051104  036440
007702  000040
007704  045522  033060  030055  RKVADR: .ASCIZ  /RK06-07 VEC ADR = /
007712  020067  042526  020103
007720  042101  020122  020075
007726     000
007727     122  030113  026466  RKPRTY: .ASCIZ  /RK06-07 PRIORITY =/
007734  033460  050040  044522
007742  051117  052111  020131
007750  000075
```

```
3273  007752  052012  050131  020105  TYPSYS: .ASCIZ  <12>/TYPE NAME OF NEXT SUBSYS TO TEST (A-P) : /
3274  007760  040516  042515  047440
3275  007766  020106  042516  052130
3276  007774  051440  041125  054523
3277  010002  020123  047524  052040
3278  010010  051505  020124  040450
3279  010016  050055  020051  020072
3280  010024     000
3281  010025     012  051104  053111  ASKTYP: .ASCIZ  <12>/DRIVE TYPE  6<CR> FOR RK06, 7<CR> FOR RK07: /
3282  010032  020105  054524  042520
3283  010040  020040  036066  051103
3284  010046  020076  047506  020122
3285  010054  045522  033060  020054
3286  010062  036067  051103  020076
3287  010070  047506  020122  045522
3288  010076  033460  020072     000
3289  010103     123  041125  054523  SYSDRV: .ASCIZ  /SUBSYS   DRIVE(S) = /
3290  010110  020123  020040  051104
3291  010116  053111  024105  024523
3292  010124  036440  000040
3293  010130  044527  046114  052040  WILTST: .ASCIZ  /WILL TEST DRIVE(S) /
3294  010136  051505  020124  051104
3295  010144  053111  024105  024523
3296  010152  000040
3297  010154  047440  020116  052523  ONSUBS: .ASCIZ  / ON SUBSYS  /<15><12><12>
3298  010162  051502  051531  020040
3299  010170  005015  000012
3300  010174  051511  052040  042510  ANOTHR: .ASCIZ  /IS THERE ANOTHER SUBSYS (Y OR N <CR>) ? /
3301  010202  042522  040440  047516
3302  010210  044124  051105  051440
3303  010216  041125  054523  020123
3304  010224  054450  047440  020122
3305  010232  020116  041474  037122
3306  010240  020051  020077     000
3307  010245     012  047515  042522  DROVFL: .ASCIZ  <12>/MORE THAN 16 DRIVES ENTERED !/<15><12><12>
3308  010252  052040  040510  020116
3309  010260  033061  042040  044522
3310  010266  042526  020123  047105
3311  010274  042524  042522  020104
3312  010302  006441  005012     000
3313  010307     012  025052  051440  SRPASS: .ASCIZ  <12>/** STARTING PASS    **/<15><12>
3314  010314  040524  052122  047111
3315  010322  020107  040520  051523
3316  010330  020040  020040  006452
3317  010336  000012
3318  010340  046412  052517  052116  MNTPAK: .ASCIZ  <12>/MOUNT PACK ON DRIVE    AND LOAD./<15><12>
3319  010346  050040  041501  020113
3320  010354  047117  042040  044522
3321  010362  042526  020040  020040
3322  010370  047101  020104  047514
3323  010376  042101  006456  000012
3324  010404  052412  046116  040517  DISPAK: .ASCIZ  <12>/UNLOAD DRIVE    AND REMOVE PACK./<15><12>
3325  010412  020104  051104  053111
3326  010420  020105  020040  040440
3327  010426  042116  051040  046505
3328  010434  053117  020105  040520
```

```
 3329  010442  045503  006456  000012
 3330  010450  051412  040524  052122  STR204: .ASCIZ  <12>/START AT ADR 204 FOR PASS 1 ON SUBSYS  ./<15><12><12>
 3331  010456  040440  020124  042101
 3332  010464  020122  030062  020064
 3333  010472  047506  020122  040520
 3334  010500  051523  030440  047440
 3335  010506  020116  052523  051502
 3336  010514  051531  020040  006456
 3337  010522  005012     000
 3338  010525     012  052123  051101  STR220: .ASCIZ  <12>/START AT ADR 220 FOR PASS 2 ON SUBSYS  ./<15><12><12>
 3339  010532  020124  052101  040440
 3340  010540  051104  031040  030062
 3341  010546  043040  051117  050040
 3342  010554  051501  020123  020062
 3343  010562  047117  051440  041125
 3344  010570  054523  020123  027040
 3345  010576  005015  000012
 3346  010602  054524  042520  051040  TYPRWN: .ASCIZ  /TYPE R<CR> WHEN /
 3347  010610  041474  037122  053440
 3348  010616  042510  020116     000
 3349  010623     104  044522  042526  DRIRDY: .ASCIZ  /DRIVE READY : /
 3350  010630  051040  040505  054504
 3351  010636  035040  000040
 3352  010642  047504  042516  035040  DRIDON: .ASCIZ  /DONE : /
 3353  010650  000040
 3354  010652  047516  042040  044522  NODRLS: .ASCIZ  /NO DRIVES LISTED FOR THIS SUBSYS !/<15><12>
 3355  010660  042526  020123  044514
 3356  010666  052123  042105  043040
 3357  010674  051117  052040  044510
 3358  010702  020123  052523  051502
 3359  010710  051531  020440  005015
 3360  010716     000
 3361  010717     015  051412  047503  RESLTS: .ASCII  <15><12>/SCORES FOR DRIVE     (0-12 DEC.) :/<15><12><12>
 3362  010724  042522  020123  047506
 3363  010732  020122  051104  053111
 3364  010740  020105  020040  020040
 3365  010746  030050  030455  020062
 3366  010754  042504  027103  020051
 3367  010762  006472  005012
 3368  010766  051124  041501  004513          .ASCII  /TRACK  DRIVE   OVRWRT  OVRWRT  READ    READ/<15><12>
 3369  010774  051104  053111  004505
 3370  011002  053117  053522  052122
 3371  011010  047411  051126  051127
 3372  011016  004524  042522  042101
 3373  011024  051011  040505  006504
 3374  011032     012
 3375  011033     116  046525  051011          .ASCII  /NUM    READ    SCORE   SCORE   SCORE   SCORE/<15><12>
 3376  011040  040505  004504  041523
 3377  011046  051117  004505  041523
 3378  011054  051117  004505  041523
 3379  011062  051117  004505  041523
 3380  011070  051117  006505     012
 3381  011075     011  047411  051506          .ASCIZ  /               OFST-   OFST+   OFST-   OFST+/<15><12><12>
 3382  011102  026524  047411  051506
 3383  011110  025524  047411  051506
 3384  011116  026524  047411  051506
```

```
3385  011124  025524  005015  000012
3386  011132  047440  051126  051127  OAVERG: .ASCIZ  / OVRWRT AVG = /
3387  011140  020124  053101  020107
3388  011146  020075     000
3389  011151     040  042522  042101  RAVERG: .ASCIZ  / READ AVG   = /
3390  011156  040440  043526  020040
3391  011164  036440  000040
3392  011170  005015  025012  020052  ENDTST: .ASCIZ  <15><12><12>/**  END OF TESTING  **/<15><12>
3393  011176  042440  042116  047440
3394  011204  020106  042524  052123
3395  011212  047111  020107  025040
3396  011220  006452  000012
3397  011224  020040  000011          TRKBUF: .ASCIZ  /       /
3398  011230  020040  004440     000  DRVBUF: .ASCIZ  /       /
3399  011235     123  046105  004506  SELF:   .ASCIZ  /SELF   /
3400  011242     000
3401  011243     011     000          TAB:    .ASCIZ  /       /
3402  011245     123  051127  036440  SWRMSG: .ASCIZ  /SWR = /
3403  011252  000040
3404  011254  020040  042516  020127  NEWMSG: .ASCIZ  /  NEW = /
3405  011262  020075     000
3406  011265     015  042012  044522  DRVSEQ: .ASCIZ  <15><12>/DRIVE(S) = /
3407  011272  042526  051450  020051
3408  011300  020075     000
3409  011303     104  044522  042526  BADDRV: .ASCIZ  /DRIVE   /
3410  011310  020040  000040
3411  011314  047516  026516  054105  NXDRIV: .ASCIZ  /NON-EXISTENT/<15><12>
3412  011322  051511  042524  052116
3413  011330  005015     000
3414  011333     116  052117  051040  NTREDY: .ASCIZ  /NOT READY/<15><12>
3415  011340  040505  054504  005015
3416  011346     000
3417  011347     127  044522  042524  WRTLOK: .ASCIZ  /WRITE-LOCKED/<15><12>
3418  011354  046055  041517  042513
3419  011362  006504  000012
3420  011366  047514  042101  042105  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
3421  011374  053440  052111  020110
3422  011402  046101  043511  020116
3423  011410  040520  045503  005015
3424  011416     000
3425  011417     111  020106  054130  REPLPK: .ASCIZ  /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", &  REPLACE IT : /
3426  011424  050104  050040  041501
3427  011432  020113  047117  042040
3428  011440  053122  030040  020054
3429  011446  054524  042520  021040
3430  011454  020131  041474  037122
3431  011462  026042  023040  020040
3432  011470  042522  046120  041501
3433  011476  020105  052111  035040
3434  011504  000040
3435  011506  005015  025012  020052  NODRTS: .ASCII  <15><12><12>/** NO DRIVES TO TEST/<15><12>
3436  011514  047516  042040  044522
3437  011522  042526  020123  047524
3438  011530  052040  051505  006524
3439  011536     012
3440  011537     120  042522  051523  CNTRDY: .ASCIZ  /PRESS "CONT" WHEN RDY/<15><12>
```

```
3441  011544  021040  047503  052116
3442  011552  020042  044127  047105
3443  011560  051040  054504  005015
3444  011566     000
3445  011567     110  046101  020124  HLTRQD: .ASCIZ  /HALT REQUESTED/<15><12>
3446  011574  042522  052521  051505
3447  011602  042524  006504  000012
3448  011610  051104  053111  020105  DROXDP: .ASCIZ  /DRIVE 0 IS LOAD MEDIUM/<15><12>
3449  011616  020060  051511  046040
3450  011624  040517  020104  042515
3451  011632  044504  046525  005015
3452  011640     000
3453  011641     015  025012  020052  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRIVE - 20 ERRORS/<15><12>
3454  011646  042040  047522  050120
3455  011654  047111  020107  051104
3456  011662  053111  020105  020055
3457  011670  030062  042440  051122
3458  011676  051117  006523  000012
3459  011704  005015  052012  051505  TSTDRN: .ASCII  <15><12><12>/TESTING DRIVE /
3460  011712  044524  043516  042040
3461  011720  044522  042526     040
3462  011725     040  005015     000  DRVNO:  .ASCIZ  / /<15><12>
3463  011731     104  044522  042526  DRIV:   .ASCIZ  /DRIVE /
3464  011736  000040
3465  011740  040503  052122  020056  CART:   .ASCIZ  /CART. /
3466  011746     000
3467  011747     123  051105  020056  SERNM:  .ASCIZ  /SER. NO.  /
3468  011754  047516  020056  000040
3469  011762  005015  040506  052103  FACTBS: .ASCIZ  <15><12>/FACTORY /
3470  011770  051117  020131     000  SOFTBS: .ASCII  <12>/SOFTWARE /
3471  011775     012  047523  052106
3472  012002  040527  042522     040  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
3473  012007     102  042101  051440
3474  012014  041505  047524  051522
3475  012022  035040  005015     000
3476  012027     040  047040  047117  NOFALS: .ASCIZ  /  NONE/
3477  012034  000105
3478
3479  012036  025052  020040  040503  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRACK!/<15><12>
3480  012044  047116  052117  051040
3481  012052  040505  020104  040502
3482  012060  020104  042523  052103
3483  012066  051117  052040  040522
3484  012074  045503  006441  000012
3485  012102  041536  005015     000  CNTRLC: .ASCIZ  /↑C/<15><12>
3486  012107     136  006532  000012  CNTRLZ: .ASCIZ  /↑Z/<15><12>
3487  012114  051136  005015     000  CNTRLR: .ASCIZ  /↑R/<15><12>
3488  012121     136  006525  000012  CNTRLU: .ASCIZ  /↑U/<15><12>
3489  012126  043536  005015     000  CNTRLG: .ASCIZ  /↑G/<15><12>
3490  012133     015  005012     000  CR2LF:  .ASCIZ  <15><12><12>
3491  012137     134     000          BKSLSH: .ASCIZ  /\/
3492  012141     054     000          COMMA:  .ASCIZ  /,/
3493  012143     040     040          SPACE6: .ASCII  /      /
3494  012145     040                  SPACE4: .ASCII  /    /
3495  012146     040                  SPACE3: .ASCII  /   /
3496  012147     040                  SPACE2: .ASCII  /  /
```

```
3497  012150  000040                  SPACE1: .ASCIZ  / /
3498                                          .EVEN
3499  012152  020040  000075          PRMBUF: .ASCIZ  /  =/
3500  012156  047527  042122  000040  WORDSP: .ASCIZ  /WORD /
3501  012164  036440  000040          EQUALS: .ASCIZ  / = /
3502  012170  020052     000          PROMPT: .ASCIZ  /* /
3503  012173     076  000040          PRMPSP: .ASCIZ  /) /
3504
3505                                          .EVEN
3506
3507
3508
3509
3510  012176  105037  003126          DFSTRT: CLRB    MDFLAG          ;SET FLAG FOR 200 START
3511  012202  000403                          BR      P1SET           ;CONTINUE
3512
3513  012204  112737  000001  003126  P1STRT: MOVB    #1,MDFLAG       ;SET FLAG FOR 204 START
3514  012212  112737  000061  003155  P1SET:  MOVB    #'1,PASSNO      ;SET PASS NO. = 1
3515  012220  000406                          BR      CMSTRT          ;CONTINUE
3516
3517  012222  112737  000002  003126  P2STRT: MOVB    #2,MDFLAG       ;SET FLAG FOR 220 START
3518  012230  112737  000062  003155          MOVB    #'2,PASSNO      ;SET PASS NO. = 2
3519
3520  012236  012737  000340  177776  CMSTRT: MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
3521                                  .SBTTL  INITIALIZE THE COMMON TAGS
3522                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3523  012244  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3524  012250  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
3525  012252  022706  001140                  CMP     #SWR,R6 ;;DONE?
3526  012256  001374                          BNE     .-6             ;;LOOP BACK IF NO
3527  012260  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
3528                                  ;;INITIALIZE A FEW VECTORS
3529  012264  012737  044606  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3530  012272  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
3531  012300  012737  044102  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3532  012306  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
3533  012314  012737  045320  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3534  012322  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
3535  012330  012737  044452  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3536  012336  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
3537  012344  013737  017250  017242          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
3538  012352  005037  001316                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3539  012356  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
3540  012364  012737  012364  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3541  012372  012737  012372  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
3542                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3543                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
3544  012400  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
3545  012404  012737  012440  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
3546  012412  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
3547  012420  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
3548  012426  022777  177777  166504          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
3549  012434  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3550                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
3551  012436  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
3552  012440  012716  012446          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
```

```
3553  012444  000002                        RTI
3554  012446  012737  000176  001140  65$:  MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
3555  012454  012737  000174  001142        MOV     #DISPREG,DISPLAY
3556  012462  012637  000004          66$:  MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
3557
3558  012466  005037  001336                CLR     $PASS           ;;CLEAR PASS COUNT
3559  012472  132737  000200  001351        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
3560  012500  001403                        BEQ     67$             ;;YES,USE NON-APT SWITCH
3561  012502  012737  001352  001140        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
3562  012510                          67$:
3563  012510  000005                        RESET                   ;CLEAR THE UNIBUS
3564  012512  012737  000006  000004        MOV     #6,@#ERRVEC     ;SET TIME-OUT VECTOR
3565  012520  005037  000006                CLR     @#ERRVEC+2
3566  012524  105037  003143                CLRB    TSTTYP          ;CLEAR OFFSET TEST IDENTIFIER
3567  012530  105037  003137                CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
3568  012534  105037  003135                CLRB    ERRCNT          ;CLEAR ERROR RETRY COUNT
3569  012540  105037  001102                CLRB    $TSTNM          ;SET TEST NO. = 0
3570  012544  112737  000001  003152        MOVB    #1,HLPOVL       ;SET HLP FILE OVLD INDICTR
3571  012552  104401  007572                TYPE    ,DZR6Q          ;TYPE PROGRAM I.D.
3572  012556  105737  003153                TSTB    NOTYPE          ;SEE IF OPERATOR NOTE SHOULD BE TYPED
3573  012562  001004                        BNE     40$             ;BR IF NOT
3574  012564  104401  054342                TYPE    ,NOTMSG         ;TYPE OPERATOR NOTE
3575  012570  105237  003153                INCB    NOTYPE          ;SET FLAG FOR NEXT TIME
3576  012574  012737  020634  000060  40$:  MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
3577  012602  012737  000200  000062        MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
3578  012610  013701  003050                MOV     RKVEC,R1        ;ADDR. OF RK06 VECTOR STORAGE
3579  012614  012721  034722                MOV     #I.INTR,(R1)+   ;SET IT TO RK06 HANDLER
3580  012620  013711  003052                MOV     RKPRI,(R1)      ;SET RK06 PRIORITY
3581  012624  105037  003127                CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
3582  012630  012737  000000  177776        MOV     #PR0,@#PS       ;ALLOW ALL INTERRUPTS AGAIN
3583  012636  012701  005162                MOV     #PRVCMD,R1      ;ZERO OUT PREVIOUS COMMAND
3584  012642  012700  000006                MOV     #6,R0           ;SET COUNTER
3585  012646  005021                  42$:  CLR     (R1)+           ;ZERO A WORD
3586  012650  005300                        DEC     R0
3587  012652  001375                        BNE     42$             ;BR IF NOT DONE YET
3588  012654  005037  005432                CLR     STALLS          ;DON'T ALLOW STALLS YET
3589  012660  004737  021136                JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
3590  012664  012737  176543  042350  44$:  MOV     #176543,$HINUM  ;INIT. PSEUDO-RANDOM NOS.
3591  012672  012737  123456  042352        MOV     #123456,$LONUM
3592  012700  004737  024176                JSR     PC,GETTYP       ;SEE IF RK06 OR RK07 & SETUP
3593  012704  105737  003126                TSTB    MDFLAG          ;SEE IF DEFAULT MODE
3594  012710  001034                        BNE     ALLSYS          ;BR IF NOT DEFAULT MODE
3595  012712  013700  001400                MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
3596  012716  013737  001404  003046        MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
3597  012724  132737  000200  001351        BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
3598  012732  001005                        BNE     18$             ;BR IF NO SIZING
3599  012734  012700  120210                MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
3600  012740  012737  177440  003046        MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
3601  012746  110037  003050          18$:  MOVB    R0,RKVEC        ;STORE VECTOR
3602  012752  105037  003051                CLRB    RKVEC+1         ;CLEAR HI BYTE
3603  012756  000300                        SWAB    R0              ;GET PRTY INTO BITS 5-7
3604  012760  042700  177437                BIC     #177437,R0      ;CLEAR OTHER BITS
3605  012764  010037  003052                MOV     R0,RKPRI        ;STORE RK06 PRIORITY
3606  012770  112737  000101  005510        MOVB    #'A,SUBSYS      ;SET SUBSYSTEM NAME = A
3607  012776  000137  013774                JMP     DFLTDR          ;GO CHECK ALL DRIVES
3608
```

```
                                        ;;*****************************************************************
                                        ;*
                                        .SBTTL  TEST 0     PARAMETER INPUT ROUTINES
                                        ;*
                                        ;;*****************************************************************
                                        ;READ AND STORE ALL SUBSYSTEMS AND DRIVES TO TEST
013002  104401  001325          ALLSYS: TYPE    ,$CRLF                  ;TYPE <CR>,<LF>
013006  105037  001102                  CLRB    $TSTNM                  ;SET TEST NO. = 0
013012  112703  000101                  MOVB    #'A,R3                  ;INIT SUBSYS INDICATOR
013016  012701  005512                  MOV     #DRVLST,R1              ;GET DRIVE LIST ADDRESS
013022  012700  000021                  MOV     #17.,R0                 ;CLEAR OUT THE DRIVE LIST
013026  005021                  2$:     CLR     (R1)+
013030  005300                          DEC     R0
013032  001375                          BNE     2$
013034  012704  005512                  MOV     #DRVLST,R4              ;GET STARTING ADRS OF DRIVE LIST
                                        ;ASK FOR DRIVES TO TEST ON THIS SUBSYSTEM
013040  110337  010112          ALLSY1: MOVB    R3,SYSDRV+7             ;SET SUBSYS NAME FOR PRINTOUT
013044  104401  010103          6$:     TYPE    ,SYSDRV                 ;ASK FOR DRIVES ON THIS SUBSYSTEM
013050  004737  023424                  JSR     PC,RDCHRS               ;READ INPUT STRING
013054  013002                          ALLSYS                          ;(↑C) RETURN ADDRESS
013056  013002                          ALLSYS                          ;(↑Z) RETURN ADDRESS
013060  013044                          6$                              ;(↑U) RETURN ADDRESS
013062  005700                          TST     R0                      ;SEE IF NULL INPUT
013064  001767                          BEQ     6$                      ;BR TO ASK AGAIN
013066  022700  000017                  CMP     #15.,R0                 ;SEE IF TOO MANY CHARS TYPED
013072  002005                          BGE     12$                     ;BR IF NOT TOO MANY
013074  104401  005212          10$:    TYPE    ,BUFFO                  ;ECHO BAD INPUT
013100  104401  001324                  TYPE    ,$QUES                  ;TYPE <?> AND <CR>,<LF>
013104  000757                          BR      6$                      ;GO ASK AGAIN
013106  005001                  12$:    CLR     R1                      ;CLEAR CHAR POINTER
013110  126127  005212  000060  14$:    CMPB    BUFFO(R1),#'0           ;SEE IF DRIVE NO. < 0
013116  103766                          BLO     10$                     ;BR TO ECHO BAD INPUT
013120  126127  005212  000067          CMPB    BUFFO(R1),#'7           ;SEE IF CHAR > 7
013126  101362                          BHI     10$                     ;BR TO ECHO BAD INPUT
013130  005201                          INC     R1                      ;INCR CHAR POINTER
013132  020100                          CMP     R1,R0                   ;SEE IF MORE CHARS TO CHECK
013134  001406                          BEQ     16$                     ;BR IF ALL CHARS CHECKED
013136  126127  005212  000054          CMPB    BUFFO(R1),#',           ;SEE IF THIS IS A COMMA
013144  001353                          BNE     10$                     ;BR IF NOT A COMMA
013146  005201                          INC     R1                      ;INCR CHAR POINTER
013150  000757                          BR      14$                     ;BR TO CONTINUE CHECKING CHARS
                                        ;PUT DRIVES FOR THIS SUBSYSTEM INTO SCRATCH LIST
013152  012701  005556          16$:    MOV     #SCRLST,R1              ;GET SCRATCH DRIVE LIST ADRS
013156  005021                          CLR     (R1)+                   ;CLEAR OUT SCRATCH DRIVE LIST
013160  005021                          CLR     (R1)+
013162  005021                          CLR     (R1)+
013164  005011                          CLR     (R1)
013166  005000                          CLR     R0                      ;INIT CHAR POINTER
013170  116001  005212          18$:    MOVB    BUFFO(R0),R1            ;GET A DRIVE NO.
013174  042701  000060                  BIC     #'0,R1                  ;STRIP ASCII BITS
013200  116061  005212  005556          MOVB    BUFFO(R0),SCRLST(R1)    ;SET ASCII DRV NO.
013206  062700  000002                  ADD     #2,R0                   ;POINT TO NEXT DRIVE NO.
013212  105760  005211                  TSTB    BUFFO-1(R0)             ;SEE IF NULL CHAR YET
013216  001364                          BNE     18$                     ;BR IF NOT DONE YET
```

```
3665                                        ;LOAD DRIVES FROM SCRATCH LIST INTO TOTAL DRIVE LIST
3666  013220  005000                        ALLSY3: CLR     R0              ;INIT SCRATCH DRV LIST POINTER
3667  013222  116014  005556                ALLSY2: MOVB    SCRLST(R0),(R4) ;MOVE THIS DRIVE INTO ACTUAL LIST
3668  013226  001411                              BEQ     22$             ;BR IF DRIVE NOT MARKED
3669  013230  005204                              INC     R4              ;INCR DRIVE LIST POINTER
3670  013232  110324                              MOVB    R3,(R4)+        ;PUT SUBSYS LETTER INTO HI BYTE
3671  013234  020427  005554                      CMP     R4,#DRVLST+34.  ;SEE IF DRIVE LIST OVERFLOW
3672  013240  103404                              BLO     22$             ;BR IF NO OVERFLOW YET
3673  013242  104401  010245                      TYPE    ,DROVFL         ;TYPE "MORE THAN 16 DRIVES ENTERED !"
3674  013246  000137  013002                      JMP     ALLSYS          ;GO START OVER, ASKING FOR DRIVES
3675  013252  005200                      22$:    INC     R0              ;INCR SCRATCH DRIVE LIST POINTER
3676  013254  020027  000010                      CMP     R0,#8.          ;SEE IF DONE WITH THIS SUBSYS
3677  013260  103760                              BLO     ALLSY2          ;BR IF NOT YET
3678                                        ;ECHO DRIVES TO TEST ON THIS SUBSYSTEM
3679  013262  104401  010130                      TYPE    ,WILTST         ;TYPE "WILL TEST DRIVE(S) "
3680  013266  005037  005444                      CLR     SCRACH          ;INIT CHAR OUTPUT BUF
3681  013272  005000                              CLR     R0              ;INIT SCRATCH LIST POINTER
3682  013274  005001                              CLR     R1              ;INIT COUNT OF LISTED DRIVES FOR SUBSYS
3683  013276  116037  005556  005444  23$:     MOVB    SCRLST(R0),SCRACH ;SEE IF THIS DRIVE LISTED
3684  013304  001407                              BEQ     26$             ;BR IF DRIVE NOT LISTED
3685  013306  005701                              TST     R1              ;SEE IF FIRST TIME HERE
3686  013310  001402                              BEQ     24$             ;BR IF FIRST TIME
3687  013312  104401  012141                      TYPE    ,COMMA          ;TYPE A COMMA
3688  013316  104401  005444              24$:    TYPE    ,SCRACH         ;TYPE A DRIVE NO.
3689  013322  005201                              INC     R1              ;INCR. COUNT OF LISTED DRIVES FOR THIS SUBSYS
3690  013324  005200                      26$:    INC     R0              ;INCR LIST POINTER
3691  013326  020027  000010                      CMP     R0,#8.          ;SEE IF DONE CHECKING LIST YET
3692  013332  002761                              BLT     23$             ;BR IF NOT DONE YET
3693  013334  110337  010167                      MOVB    R3,ONSUBS+11.   ;SET SUBSYS NO. IN MSG
3694  013340  104401  010154                      TYPE    ,ONSUBS         ;TYPE "ON SUBSYSTEM "
3695  013344  005203                              INC     R3              ;INCR SUBSYSTEM NAME
3696  013346  020327  000120                      CMP     R3,#'P          ;SEE IF NAME > P
3697  013352  101035                              BHI     ASKSYS          ;BR IF YES
3698  013354  105737  003126                      TSTB    MDFLAG          ;SEE IF 200 START
3699  013360  001002                              BNE     28$             ;BR IF  NOT
3700  013362  000137  013734                      JMP     DRVTST          ;START PASS 1
3701                                        ;ASK IF THERE IS ANOTHER SUBSYSTEM TO SPECIFY
3702  013366  104401  010174              28$:    TYPE    ,ANOTHR         ;ASK FOR ANOTHER SUBSYS
3703  013372  004737  023424                      JSR     PC,RDCHRS       ;READ RESPONSE
3704  013376  013002                              ALLSYS                  ;(↑C) RETURN ADDRESS
3705  013400  013002                              ALLSYS                  ;(↑Z) RETURN ADDRESS
3706  013402  013366                              28$                     ;(↑U) RETURN ADDRESS
3707  013404  005700                              TST     R0              ;SEE IF NULL INPUT
3708  013406  001417                              BEQ     ASKSYS          ;BR IF NULL INPUT, TO ASSUME <N>
3709  013410  022737  000131  005212              CMP     #'Y,BUFFO       ;SEE IF <Y> TYPED
3710  013416  001002                              BNE     30$             ;BR IF NOT <Y>
3711  013420  000137  013040                      JMP     ALLSY1          ;GO ASK FOR DRIVES ON NEXT SUBSYS
3712  013424  022737  000116  005212  30$:     CMP     #'N,BUFFO       ;SEE IF <N> TYPED
3713  013432  001405                              BEQ     ASKSYS          ;BR IF <N>
3714  013434  104401  005212                      TYPE    ,BUFFO          ;ECHO BAD INPUT
3715  013440  104401  001324                      TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
3716  013444  000750                              BR      28$             ;GO ASK AGAIN
3717
3718                                        ;204 OR 220 START - GET NAME OF SUBSYSTEM TO TEST NEXT
3719  013446  105037  001102              ASKSYS: CLRB    $TSTNM          ;SET TEST NO. = 0
3720  013452  104401  007752                      TYPE    ,TYPSYS         ;ASK FOR NEXT SUBSYS TO TEST
```

```
3721  013456  004737  023424                    JSR     PC,RDCHRS       ;READ RESPONSE
3722  013462  013002                            ALLSYS                  ;(↑C) RETURN ADDRESS
3723  013464  013446                            ASKSYS                  ;(↑Z) RETURN ADDRESS
3724  013466  013446                            ASKSYS                  ;(↑O) RETURN ADDRESS
3725  013470  005700                            TST     R0              ;SEE IF NULL INPUT
3726  013472  001765                            BEQ     ASKSYS          ;BR IF NULL, TO ASK AGAIN
3727  013474  023727  005212  000101            CMP     BUFFO,#'A       ;SEE IF A CHAR < A TYPED
3728  013502  103005                            BHIS    6$              ;BR IF NOT
3729  013504  104401  005212            4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
3730  013510  104401  001324                    TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
3731  013514  000754                            BR      ASKSYS          ;GO ASK AGAIN
3732  013516  023727  005212  000120    6$:     CMP     BUFFO,#'P       ;SEE IF A CHAR > P TYPED
3733  013524  101367                            BHI     4$              ;BR IF YES
3734  013526  113737  005212  005510            MOVB    BUFFO,SUBSYS    ;STORE CURRENT SUBSYSTEM NAME
3735  013534  012700  005513                    MOV     #DRVLST+1,R0    ;SET DRIVE LIST POINTER
3736  013540  122037  005510            8$:     CMPB    (R0)+,SUBSYS    ;SEE IF A DRIVE LISTED FOR THIS SUBSYS
3737  013544  001410                            BEQ     9$              ;BR IF YES
3738  013546  005200                            INC     R0              ;INCR POINTER
3739  013550  020027  005553                    CMP     R0,#DRVLST+33.  ;SEE IF WHOLE LIST CHECKED YET
3740  013554  103771                            BLO     8$              ;BR IF NOT YET
3741  013556  104401  010652                    TYPE    ,NODRLS         ;TYPE "NO DRIVES LISTED FOR SUBSYS"
3742  013562  000137  013446                    JMP     ASKSYS          ;GO  ASK FOR  A SUBSYS AGAIN
3743  013566  162700  000002            9$:     SUB     #2,R0           ;POINT TO LOW BYTE OF LIST ENTRY
3744  013572  010037  005554                    MOV     R0,DRVPTR       ;STORE POINTER
3745                                    ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
3746  013576  013746  003046                    MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
3747  013602  104401  007660                    TYPE    ,RKBADR         ;TYPE "RK06 BUS ADR = "
3748  013606  004737  021026                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
3749  013612  012637  003046                    MOV     (SP)+,RKBAS     ;STORE NEW VALUE
3750                                    ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
3751  013616  013746  003050                    MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
3752  013622  104401  007704                    TYPE    ,RKVADR         ;TYPE "RK06 VEC ADR = "
3753  013626  004737  021026                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
3754  013632  012637  003050                    MOV     (SP)+,RKVEC     ;STORE NEW VALUE
3755                                    ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
3756  013636  013746  003052            11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
3757  013642  006316                            ASL     (SP)            ;GET IT INTO BITS 0-2
3758  013644  006316                            ASL     (SP)
3759  013646  006316                            ASL     (SP)
3760  013650  000316                            SWAB    (SP)
3761  013652  104401  007727                    TYPE    ,RKPRTY         ;TYPE   "RK06 PRIORITY = "
3762  013656  004737  021026                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
3763  013662  012600                            MOV     (SP)+,R0
3764  013664  020027  000004                    CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
3765  013670  002414                            BLT     12$             ;BR IF NEW VALUE TOO SMALL
3766  013672  020027  000007                    CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
3767  013676  003011                            BGT     12$             ;BR IF NEW VALUE TOO LARGE
3768  013700  000300                            SWAB    R0              ;GET PRIORITY INTO BITS 5-7
3769  013702  006200                            ASR     R0
3770  013704  006200                            ASR     R0
3771  013706  006200                            ASR     R0
3772  013710  010037  003052                    MOV     R0,RKPRI        ;STORE NEW VALUE
3773  013714  104401  001325                    TYPE    ,$CRLF
3774  013720  000405                            BR      DRVTST
3775  013722  104401  005212           12$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
3776  013726  104401  001324                    TYPE    ,$QUES
```

```
3777  013732  000741                          BR      11$             ;GO ASK AGAIN
3778
3779  013734  113737  003155  010331  DRVTST: MOVB    PASSNO,SRPASS+18. ;SET PASS NO. FOR PRINTOUT
3780  013742  104401  010307          TYPE    ,SRPASS         ;TYPE "STARTING PASS X"
3781  013746  122737  000061  003155  CMPB    #'1,PASSNO      ;SEE IF PASS 1
3782  013754  001002                  BNE     2$              ;BR IF NOT PASS 1
3783  013756  000137  014346          JMP     TST1            ;START PASS 1
3784  013762  112737  000004  001102  2$:     MOVB    #4,$TSTNM       ;SET TEST NUMBER = 4
3785  013770  000137  015146          JMP     TST5            ;START PASS 2
3786
3787
3788                                  ;ADR 200 START - COMPILE SCRATCH LIST OF DRIVES TO SIZE
3789  013774  005000                  DFLTDR: CLR     R0              ;CLEAR DRIVE NUMBER
3790  013776  013703  001406          MOV     $DEVM,R3        ;GET APT DEVICE MAP
3791  014002  132737  000200  001351  BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
3792  014010  001002                  BNE     1$              ;BR IF NO SIZING
3793  014012  012703  000377          MOV     #377,R3         ;SET UP FOR SIZING
3794  014016  012701  000001  1$:     MOV     #BIT0,R1        ;SET BIT POINTER
3795  014022  105060  005556  2$:     CLRB    SCRLST(R0)      ;INITIALIZE DRIVE ENTRY
3796  014026  030103                  BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
3797  014030  001405                  BEQ     4$              ;BR IF NOT
3798  014032  110060  005556          MOVB    R0,SCRLST(R0)   ;MARK DRIVE IN SCRATCH LIST
3799  014036  152760  000060  005556  BISB    #'0,SCRLST(R0)  ;CONVERT NO. TO ASCII
3800  014044  006301          4$:     ASL     R1              ;SHIFT BIT POINTER
3801  014046  005200                  INC     R0              ;INCREMENT DRIVE NUMBER
3802  014050  022700  000010          CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
3803  014054  001362                  BNE     2$              ;BR IF NOT DONE YET
3804  014056  132737  000200  001351  BITB    #BIT7,$ENVM     ;SEE IF PROGRAM SHOULD SIZE
3805  014064  001416                  BEQ     8$              ;BR IF YES
3806  014066  012704  005512  10$:     MOV     #DRVLST,R4      ;GET DRIVE LIST ADDRESS
3807  014072  113703  005510          MOVB    SUBSYS,R3       ;GET SUBSYS NAME IN R3
3808  014076  010401                  MOV     R4,R1           ;GET DRIVE LIST ADRS
3809  014100  010437  005554          MOV     R4,DRVPTR       ;SET POINTER TO FIRST DRIVE TO TEST
3810  014104  012700  000021          MOV     #17.,R0         ;SET COUNTER
3811  014110  005021          11$:    CLR     (R1)+           ;CLEAR A DRIVE LIST WORD
3812  014112  005300                  DEC     R0              ;DECREMENT COUNTER
3813  014114  001375                  BNE     11$             ;BR IF NOT DONE YET
3814  014116  000137  013220          JMP     ALLSY3          ;GO MOVE SCRATCH LIST INTO LIST
3815                                  ;REMOVE NON-EXISTENT DRIVES FROM SCRATCH LIST
3816  014122  005000          8$:     CLR     R0              ;CLEAR DRIVE NUMBER
3817  014124  105760  005556  12$:    TSTB    SCRLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN SCRATCH LIST
3818  014130  001475                  BEQ     14$             ;BR IF NOT MARKED
3819  014132  010037  005430          MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
3820  014136  004737  021170          JSR     PC,INITSS       ;CLEAR DRIVER AND INIT S.S.
3821  014142  042712  000100  3$:     BIC     #IE,(R2)        ;DISABLE RK06 INTERRUPT
3822  014146  113737  005430  000010  MOVB    DRIVE,RKCS2
3823  014154  105737  003133          TSTB    TYPFMT          ;SEE IF RK07
3824  014160  001003                  BNE     5$              ;BR IF YES
3825  014162  012712  000001          MOV     #GO,(R2)        ;ELSE SET GO BIT FOR RK06 IN RKCS1
3826  014166  000402                  BR      7$
3827  014170  012712  002001  5$:     MOV     #<CDT!GO>,(R2)  ;ELSE SET GO FOR RK07 IN RKCS1
3828  014174  005037  005444  7$:     CLR     SCRACH          ;CLEAR STALL CTR
3829  014200  005237  005444  15$:    INC     SCRACH          ;INCREMENT STALL COUNTER
3830  014204  001375                  BNE     15$             ;STALL FOR SEVERAL MILLI-SEC
3831  014206  032762  001000  000010  BIT     #MDS,RKCS2(R2)  ;SEE IF MDS ERROR
3832  014214  001417                  BEQ     18$             ;BR IF NOT
```

```
3833  014216  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
3834  014224  011265  000016                  MOV     (R2),P.CSI(R5)  ;GET RKCS1 FOR REPORT
3835  014230  004737  037644                  JSR     PC,I.CST1       ;GET OTHER RK611 REGS FOR REPORT
3836  014234  004737  031224                  JSR     PC,REPSUP       ;SET UP ERROR REPORT
3837  014240  104052                          ERROR   52              ;REPORT MDS ERROR
3838  014242  052737  000200  005424          BIS     #ABORT,RECODE   ;SET ABORT FLAG
3839  014250  000137  033336                  JMP     ALLTRM          ;GO ABORT TESTING
3840  014254  004737  021170          18$:    JSR     PC,INITSS       ;INIT THE S.S.
3841  014260  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
3842  014266  012737  020500  003056          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADR
3843  014274  012737  000377  005476          MOV     #377,NEWON      ;INIT ONLINE INDICATOR
3844  014302  004737  030146                  JSR     PC,DRVCAL       ;READ DRIVE STATUS
3845  014306  012737  031504  003056          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNORMAL RETURN ADDRESS
3846  014314  022737  000377  005476          CMP     #377,NEWON      ;SEE IF THIS DRIVE EXISTS
3847  014322  001006                          BNE     16$             ;BR IF NO
3848  014324  005200                  14$:    INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
3849  014326  022700  000010                  CMP     #10,R0          ;SEE IF DONE CHECKING LIST
3850  014332  001274                          BNE     12$             ;BR IF NOT DONE YET
3851  014334  000137  014066                  JMP     10$             ;GO BACK TO LIST SELECTED DRIVES
3852  014340  105060  005556          16$:    CLRB    SCRLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
3853  014344  000767                          BR      14$             ;CONTINUE CHECKING THE LIST
3854
3855
3856
3857
3858
3859                                  ;;*****************************************************************
3860                                  ;*TEST 1         MOUNTING OF TEST CARTRIDGE FOR PASS 1
3861                                  ;*
3862                                  ;*       THE PROGRAM TYPES "STARTING PASS 1". THEN, THE OPERATOR
3863                                  ;*       MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
3864                                  ;*       AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
3865                                  ;*       FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
3866                                  ;*       DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
3867                                  ;*       ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
3868                                  ;*       IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS.
3869                                  ;*
3870                                  ;;*****************************************************************
3871  014346  000004                  TST1:   SCOPE
3872  014350  012737  000001  001334          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3873  014356  004737  022042                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
3874  014362  004737  021170                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3875  014366  004737  020774                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
3876  014372  004737  022110                  JSR     PC,MTCART
3877
3878
3879
3880
3881
3882                                  ;;*****************************************************************
3883                                  ;*TEST 2         BASIC READ/WRITE DATA TEST
3884                                  ;*
3885                                  ;*       THE PROGRAM PERFORMS A WRITE AND WRITE CHECK OPERATION
3886                                  ;*       USING A "WORST CASE" DATA PATTERN, AT THE APPROPRIATE SECTOR
3887                                  ;*       FOR THIS DRIVE (FROM TABLE A) ON ALL SURFACES. THE PURPOSE
3888                                  ;*       OF THIS OPERATION IS TO VERIFY THE BASIC READ/WRITE
```

```
                                        T2      BASIC READ/WRITE DATA TEST

                                        ;*      CAPABILITY OF THE DRIVE ON PASS 1. THE ENTIRE SECTOR
                                        ;*      IS WRITTEN WITH THE REPETITION OF THE DATA PATTERN SHOWN
                                        ;*      IN TABLE B.
                                        ;*
                                        ;**********************************************************************
014376  000004                        TST2:   SCOPE
014400  012737  000002  001334                MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
014406  004737  022042                        JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
014412  004737  021170                        JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
014416  004737  020774                        JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
014422  112737  000001  003127                MOVB    #1,TSTING               ;ALLOW TTY ESCAPE
                                        ;LOAD RWBUF WITH DATA PATTERN
014430  004737  024502                        JSR     PC,LODSEC
                                        ;SET UP PARAMETERS
014434  113700  003154                        MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NUMBER
014440  006300                            ASL     R0              ;CONVERT IT TO INDEX
014442  006300                            ASL     R0
014444  105737  003133                    TSTB    TYPFMT          ;SEE IF RK07
014450  001011                            BNE     1$              ;BR IF YES
014452  016065  005566  000002            MOV     TABLEA(R0),P.CYLN(R5) ;GET CYL # FOR THIS DRIVE
014460  062700  000002                    ADD     #2,R0           ;INCREMENT INDEX
014464  116065  005566  000004            MOVB    TABLEA(R0),P.SECT(R5) ;GET SECTOR # FOR THIS DRIVE
014472  000410                            BR      2$

014474  016065  005666  000002    1$:     MOV     TABL7A(R0),P.CYLN(R5)   ;GET RK07 TABLE
014502  062700  000002                    ADD     #2,R0
014506  116065  005666  000004            MOVB    TABL7A(R0),P.SECT(R5)   ;GET SECTOR # FOR THE DRV

014514  012765  053342  000010    2$:     MOV     #RWBUF,P.BALO(R5)       ;SET BUS ADDRESS
014522  012765  177400  000012            MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
                                        ;WRITE THE DATA
014530  112765  000123  000001    6$:     MOVB    #WRDATA,P.CMND(R5)      ;SET WRITE COMMAND
014536  004737  030146                    JSR     PC,DRVCAL               ;WRITE THE SECTOR
                                        ;PERFORM WRITE CHECK OF SECTOR
014542  112765  000131  000001            MOVB    #WRTCHK,P.CMND(R5)      ;SET WRITE CHECK COMMAND
014550  004737  030146                    JSR     PC,DRVCAL       ;PERFORM THE WRITE CHECK
                                        ;INCREMENT TRACK ADDRESS
014554  105265  000005                    INCB    P.TRCK(R5)      ;INCREMENT TRACK NO.
014560  126527  000005  000002            CMPB    P.TRCK(R5),#2   ;SEE IF ALL SURFACES WRITTEN YET
014566  101760                            BLOS    6$              ;BR IF NOT YET


                                        ;**********************************************************************
                                        ;*TEST 3         WRITE OVRWRT AND COMPAT CYL BLOCKS FOR CURRENT DRIVE
                                        ;*      ALL SECTORS ARE WRITTEN FOR THE CURRENT DRIVE WITHIN THE
                                        ;*      CYLINDER BLOCKS IN TABLEC ON ALL SURFACES, USING A SINGLE
                                        ;*      REPEATED WORD OF THE PSEUDO-RANDOM DATA PATTERN IN TABLEG. LOGICAL DRIVE 0
                                        ;*      USES WORD 0, LOGICAL DRIVE 14 USES WORD 14, ETC.  THESE CYLINDER
                                        ;*      BLOCKS ARE THE OVERWRITE AND COMPATIBILITY TEST DATA TO
                                        ;*      BE USED IN PASS 2.
                                        ;**********************************************************************
014570  000004                        TST3:   SCOPE
014572  012737  000003  001334                MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
014600  004737  022042                        JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
```

```
3945  014604  004737  021170              JSR     PC,INIISS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
3946  014610  004737  020774              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
3947  014614  004737  022642              JSR     PC,INTBLD       ;LOAD INITIAL VALUES INTO TABLED
3948  014620  005000                      CLR     R0              ;INITIALIZE TABLEC INDEX TO 0
3949  014622  105737  003133          4$: TSTB    TYPFMT          ;SEE IF RK07
3950  014626  001013                      BNE     1$              ;BR IF YES
3951  014630  016065  006026  000002      MOV     TABLEC(R0),P.CYLN(R5) ;GET START OVRWRT CYL FROM TABLEC
3952  014636  004737  023106              JSR     PC,WRTBLK       ;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S
3953  014642  062700  000002              ADD     #2,R0           ;INCREMENT TABLEC POINTER
3954  014646  016065  006026  000002      MOV     TABLEC(R0),P.CYLN(R5) ;GET START COMPAT CYL FROM TABLEC
3955  014654  000412                      BR      2$
3956
3957  014656  016065  006060  000002  1$: MOV     TABL7C(R0),P.CYLN(R5)
3958  014664  004737  023106              JSR     PC,WRTBLK
3959  014670  062700  000002              ADD     #2,R0
3960  014674  016065  006060  000002      MOV     TABL7C(R0),P.CYLN(R5)
3961  014702  004737  023106          2$: JSR     PC,WRTBLK       ;WRITE THIS COMPAT CYL BLK ON ALL SURF'S
3962  014706  062700  000002              ADD     #2,R0           ;INCREMENT TABLEC POINTER
3963  014712  020027  000030              CMP     R0,#30          ;SEE IF ON LAST CURRENT ZONE YET
3964  014716  002003                      BGE     8$              ;BR IF YES
3965  014720  004737  023022              JSR     PC,ROTBLD       ;ROTATE TABLED SEVERAL SECTORS
3966  014724  000736                      BR      4$              ;CONTINUE WRITING BLOCKS
3967  014726  004737  022642          8$: JSR     PC,INTBLD       ;RE-LOAD INITIAL VALUES INTO TABLED
3968  014732  105737  003133              TSTB    TYPFMT          ;SEE IF RK07
3969  014736  001004                      BNE     3$              ;BR IF YES
3970  014740  016065  006026  000002      MOV     TABLEC(R0),P.CYLN(R5) ;GET START OVRWRT CYL FROM TABLEC
3971  014746  000403                      BR      5$
3972
3973  014750  016065  006060  000002  3$: MOV     TABL7C(R0),P.CYLN(R5)
3974  014756  004737  023106          5$: JSR     PC,WRTBLK       ;WRITE THIS OVRWRT CYL BLK ON ALL SURF'S
3975
3976
3977
3978                                  ;*****************************************************************
3979                                  ;*TEST 4        DISMOUNTING OF TEST CARTRIDGE IN PASS 1
3980                                  ;*
3981                                  ;*      THE PROGRAM DIRECTS THE OPERATOR TO UNLOAD THE DRIVE CURRENTLY
3982                                  ;*      UNDER TEST, AND TO REMOVE THE TEST PACK. THEN, THE PROGRAM
3983                                  ;*      DOES ONE OF 4 THINGS :
3984                                  ;*      (1) IF THERE IS ANOTHER DRIVE TO TEST ON THIS SUBSYSTEM, THE
3985                                  ;*      PROGRAM JUMPS TO TEST 1 FOR THE NEXT DRIVE.
3986                                  ;*      (2) IF THERE IS ONLY ONE SUBSYSTEM, (FROM 200 START), AND
3987                                  ;*      IF PASS 1 HAS BEEN COMPLETED ON ALL DRIVES, THE PROGRAM STARTS
3988                                  ;*      PASS 2 ON THE FIRST DRIVE, BY JUMPING TO TEST 5.
3989                                  ;*      (3) IF THERE IS ANOTHER SUBSYSTEM TO PERFORM PASS 1 ON, THE PROGRAM
3990                                  ;*      TELLS THE OPERATOR TO RESTART AT ADDRESS 204, AND THEN IT HALTS.
3991                                  ;*      (4) IF THERE ARE NO MORE DRIVES TO TEST IN PASS 1 ON ANY
3992                                  ;*      OTHER SUBSYSTEM, THE PROGRAM TELLS THE OPERATOR TO
3993                                  ;*      RESTART AT ADDRESS 220 FOR PASS 2 ON THE NEXT SUBSYSTEM, AND
3994                                  ;*      THEN IT HALTS.
3995                                  ;*
3996                                  ;*****************************************************************
3997  014762  000004              TST4:   SCOPE
3998  014764  012737  000004  001334      MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
3999  014772  004737  022042              JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4000  014776  004737  021170              JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
```

```
4001  015002  004737  020774                      JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4002  015006  105037  003127                      CLRB    TSTING          ;DON'T ALLOW TTY ESCAPE
4003  015012  004737  022472                      JSR     PC,DMCART       ;DIRECT OPERATOR TO DISMOUNT PACK
4004  015016  062737  000002  005554              ADD     #2,DRVPTR       ;INCREMENT DRIVE LIST POINTER
4005  015024  013700  005554                      MOV     DRVPTR,R0
4006  015030  005710                              TST     (R0)            ;SEE IF ALL DONE WITH PASS 1 YET
4007  015032  001420                              BEQ     8$              ;BR IF DONE WITH PASS 1
4008  015034  126037  000001  005510              CMPB    1(R0),SUBSYS    ;SEE IF DONE WITH THIS SUBSYS YET
4009  015042  001004                              BNE     4$              ;BR IF DONE WITH THIS SUBSYS
4010  015044  105037  001102                      CLRB    $TSTNM          ;CLEAR TEST NUMBER
4011  015050  000137  014346                      JMP     TST1            ;DO PASS 1 ON NEXT DRIVE OF SUBSYS
4012  015054  116037  000001  010517  4$:         MOVB    1(R0),STR204+39.  ;PUT SUBSYS NAME INTO MSG
4013  015062  104401  010450          5$:         TYPE    ,STR204         ;TYPE RESTART MSG FOR PASS 1 ON NEXT SUBSYS
4014  015066  000000                  6$:         HALT                    ;HALT THE PROCESSOR SO THAT
4015  015070  000137  000204                      JMP     204             ;  OPERATOR CAN RESTART AT PROPER ADRS
4016  015074  012737  005512  005554  8$:         MOV     #DRVLST,DRVPTR  ;RE-INIT DRIVE LIST POINTER
4017  015102  105737  003126                      TSTB    MDFLAG          ;SEE IF 200 START
4018  015106  001007                              BNE     12$             ;BR IF NOT
4019  015110  112737  000062  010331              MOVB    #'2,SRPASS+18.  ;SET PASS NO. = 2 FOR PRINTOUT
4020  015116  104401  010307                      TYPE    ,SRPASS         ;TYPE "STARTING PASS 2"
4021  015122  000137  015146                      JMP     TST5            ;GO START PASS 2
4022  015126  112737  000101  010574  12$:        MOVB    #'A,STR220+39.  ;PUT SUBSYS NAME INTO MSG
4023  015134  104401  010525                      TYPE    ,STR220         ;TYPE RESTART MSG FOR PASS 2 ON FIRST SUBSYS
4024  015140  000000                              HALT                    ;HALT PROCESSOR SO THAT OPERATOR CAN RESTART
4025  015142  000137  000220                      JMP     220             ;    AT THE PROPER ADRS
4026
4027
4028
4029                                      ;;*****************************************************************
4030                                      ;*TEST 5        MOUNTING OF TEST CARTRIDGE FOR PASS 2
4031                                      ;*
4032                                      ;*      THE PROGRAM TYPES "STARTING PASS 2". THEN, THE OPERATOR
4033                                      ;*      MOUNTS THE PACK ON THIS DRIVE AND MANUALLY LOADS THE HEADS,
4034                                      ;*      AS DIRECTED BY THE PROGRAM. IT THEN CHECKS THIS DRIVE
4035                                      ;*      FOR VALID STATUS, AND READS THE HEADER ON SECTOR 0, TO
4036                                      ;*      DETERMINE THE FORMAT. THE FORMAT IS STORED IN "FORMAT".
4037                                      ;*      ALSO, THE PROGRAM READS AND STORES THE BAD SECTOR FILES, AND
4038                                      ;*      IT TYPES THE DRIVE AND CARTRIDGE SERIAL NUMBERS (IF STARTED
4039                                      ;*      AT ADDRESS 204 OR 220).
4040                                      ;*
4041                                      ;*****************************************************************
4042  015146  000004                  TST5:       SCOPE
4043  015150  012737  000005  001334              MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4044  015156  004737  022042                      JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4045  015162  004737  021170                      JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4046  015166  004737  020774                      JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4047  015172  112737  000062  003155              MOVB    #'2,PASSNO      ;SET PASS NO. = 2
4048  015200  004737  022110                      JSR     PC,MTCART
4049
4050
4051
4052                                      ;;*****************************************************************
4053                                      ;*TEST 6        OVERWRITE TEST
4054                                      ;*
4055                                      ;*      THE PROGRAM PROCEEDS TO TEST THIS DRIVE'S OVERWRITE CAPABILITY.
4056                                      ;*      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.
                                        ;*      DRIVE 0 USES WORD 0, DRIVE 1 USES WORD 1, 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-IN., IN BOTH
                                        ;*      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.
                                        ;*
                                        ;**********************************************************************
015204  000004                    TST6:   SCOPE
015206  012737  000006  001334            MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
015214  004737  022042                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
015220  004737  021170                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
015224  004737  020774                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
015230  112737  000001  003127            MOVB    #1,TSTING       ;ALLOW TTY ESCAPE
015236  105037  003147                    CLRB    DIF100          ;CLEAR "CYL DIF = 100/200" FLAG
                                        ;OVERWRITE THE APPROPRIATE CYLS FOR THIS DRIVE
015242  112765  000117  000001            MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
015250  105065  000002                    CLRB    P.CYLN(R5)      ;SET CYL = 0
015254  004737  030146                    JSR     PC,DRVCAL       ;SEEK TO CYL 0
015260  012703  000007                    MOV     #7,R3           ;SET COUNTER = 7
015264  112765  000123  000001            MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
015272  113700  003154                    MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NUMBER
015276  010065  000002                    MOV     R0,P.CYLN(R5)   ;SET STARTING OVERWRITE CYL FOR THIS DRIVE
015302  006300                            ASL     R0              ;GET DATA PATTERN INDEX
015304  062700  006512                    ADD     #TABLEG,R0      ;GET PATTERN WORD ADRS
015310  011037  053342                    MOV     (R0),RWBUF      ;PUT DATA WORD INTO BUFFER
015314  012765  053342  000010            MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
015322  052765  100000  000014            BIS     #DTBAII,P.PRST(R5) ;SET BUS ADRS INCREMENT INHIBIT
015330  012765  137000  000012            MOV     #-16896.,P.WC(R5) ;SET 22-SECTOR WORD COUNT
015336  105737  003134                    TSTB    FORMAT          ;DETERMINE THE FORMAT
015342  001403                            BEQ     6$              ;BR IF 22 SECTORS
015344  012765  142000  000012            MOV     #-15360.,P.WC(R5) ;SET 20-SECTOR WORD COUNT
015352  004737  027500            6$:     JSR     PC,SVPRMS       ;SAVE PARAMETERS OF TRANSFER
015356  004737  027570                    JSR     PC,TRNSFR       ;WRITE THE DATA
015362  063765  024412  000002            ADD     HOLD3,P.CYLN(R5) ;INCR THE CYL NO.
015370  005303                            DEC     R3              ;SEE IF DONE WITH ALL CYLS YET
015372  001367                            BNE     6$              ;BR IF NOT YET
                                        ;READ OVERWRITE CYLS WITH RANGE OF OFFSETS
015374  112737  000001  003143            MOVB    #1,TSTTYP       ;SET INDICATOR FOR OVRWRT TEST
015402  012700  006612                    MOV     #NOSCR0,R0      ;GET STARTING ADRS OF OVRWRT SCORES
015406  005020            10$:    CLR     (R0)+           ;INIT SCORE TO 0
015410  020027  007112                    CMP     R0,#NOSCR0+192. ;SEE IF ALL SCORES CLEARED YET
015414  103774                            BLO     10$             ;BR IF NOT DONE YET
015416  012703  000007                    MOV     #7,R3           ;SET CYL CNTR = 7
015422  113700  003154                    MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NO.
```

```
4113  015426  010065  000002                  MOV     R0,P.CYLN(R5)   ;GET START OVRWRT CYL FOR THIS DRIVE
4114  015432  004737  022642          11$:    JSR     PC,INTBLD       ;INIT TABLED
4115  015436  004737  017300          12$:    JSR     PC,REDOFS       ;DO READS WITH OFFSETS ON THIS CYL
4116  015442  004737  017644                  JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUM
4117  015446  063765  024412  000002          ADD     HOLD3,P.CYLN(R5) ;INCR CYL NO.
4118  015454  112737  000001  003147          MOVB    #1,DIF100       ;SET "CYL DIF = 100/200" FLAG
4119  015462  005303                          DEC     R3              ;DECR CYL CNTR
4120  015464  004737  023022                  JSR     PC,ROTBLD       ;ROTATE TABLED FOR NEXT CURRENT ZONE
4121  015470  020327  000001                  CMP     R3,#1           ;SEE IF ON LAST ZONE YET
4122  015474  003360                          BGT     12$             ;BR IF NOT YET
4123  015476  001755                          BEQ     11$             ;BR IF ON LAST CYL
4124  015500  004737  020024                  JSR     PC,AVSCOR       ;COMPUTE AVERAGE OVERWRITE SCORES
4125  015504  105037  003143                  CLRB    TSTTYP          ;CLEAR OVRWRT TEST INDICATOR
4126
4127
4128
4129                                  ;;*****************************************************************
4130                                  ;*TEST 7         DRIVE SELF-TEST
4131                                  ;*
4132                                  ;*      THE PROGRAM EVALUATES THE DRIVE'S ABILITY TO WRITE AND READ
4133                                  ;*      ITS OWN DATA, AT VARIOUS ANGULAR POSITIONS ON THE PACK. FIRST,
4134                                  ;*      SECTORS 4,11,16, AND 23 OF THE APPROPRIATE CYLINDERS SHOWN
4135                                  ;*      IN TABLEF FOR THIS DRIVE ARE WRITTEN WITH THE DATA PATTERN
4136                                  ;*      SHOWN IN TABLEB, FOR ALL SURFACES. THEN, THE SECTORS ARE READ
4137                                  ;*      WITH THE FOLLOWING OFFSETS APPLIED : 100,200,300,400,500,600
4138                                  ;*      700,800,900,1000,1100,1200 MICRO-IN., IN (+) AND (-) DIRECTIONS.
4139                                  ;*      THE PROGRAM SCANS FOR DATA CHECK ERRORS DURING EACH READ, AND IT
4140                                  ;*      COMPUTES A SCORE WHICH IS PROPORTIONAL TO THE FAILING OFFSET
4141                                  ;*      MAGNITUDE.  THEN, THE SCORES FOR ALL SECTORS READ ARE AVERAGED,
4142                                  ;*      TO COME UP WITH A DRIVE SELF-TEST SCORE FOR EACH OFFSET
4143                                  ;*      DIRECTION, FOR EACH SURFACE.
4144                                  ;*      THIS SCORE IS SAVED FOR LATER USE, TO BECOME THE STANDARD
4145                                  ;*      FOR THE COMPATIBILITY DATA READS WHICH ARE TO FOLLOW.
4146                                  ;*
4147                                  ;;*****************************************************************
4148  015510  000004                  TST7:   SCOPE
4149  015512  012737  000007  001334          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
4150  015520  004737  022042                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4151  015524  004737  021170                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4152  015530  004737  020774                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4153  015534  005037  007412                  CLR     NSSCR0          ;INIT SELF-TEST SCORES TO 0
4154  015540  005037  007414                  CLR     NSSCR1
4155  015544  005037  007416                  CLR     NSSCR2
4156  015550  005037  007420                  CLR     PSSCR0
4157  015554  005037  007422                  CLR     PSSCR1
4158  015560  005037  007424                  CLR     PSSCR2
4159  015564  105037  003147                  CLRB    DIF100          ;CLEAR "CYL DIF = 100/200" FLAG
4160  015570  012703  000006                  MOV     #6,R3           ;SET CYL COUNTER = 6
4161  015574  113700  003154                  MOVB    LOGDRV,R0       ;GET LOGICAL DRIVE NO.
4162  015600  062700  000060                  ADD     #60,R0          ;COMPUTE STARTING CYL NO.
4163  015604  012765  053342  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADRS
4164  015612  004737  024502                  JSR     PC,LODSEC       ;LOAD R/W BUF WITH DATA PATTERN
4165  015616  012765  177400  000012          MOV     #-256.,P.WC(R5) ;SET WORD COUNT FOR 1 SECTOR
4166  015624  112765  000004  000004  3$:     MOVB    #4,P.SECT(R5)   ;SET SECTOR = 4
4167  015632  105065  000005          4$:     CLRB    P.TRCK(R5)      ;SET TRACK = 0
4168  015636  105037  007432                  CLRB    NSCOR0          ;INIT SCRATCH SCORES TO 0
```

```
                                T7      DRIVE SELF-TEST

 4169  015642  105037  007452                CLRB    NSCOR1
 4170  015646  105037  007472                CLRB    NSCOR2
 4171  015652  105037  007512                CLRB    PSCOR0
 4172  015656  105037  007532                CLRB    PSCOR1
 4173  015662  105037  007552                CLRB    PSCOR2
 4174  015666  105037  003143        6$:     CLRB    TSTTYP          ;CLEAR OFFSET TEST INDICATOR
 4175  015672  005065  000002                CLR     P.CYLN(R5)      ;SET CYL = 0
 4176  015676  112765  000117  000001        MOVB    #SEEK,P.CMND(R5)        ;SET SEEK COMMAND
 4177  015704  004737  030146                JSR     PC,DRVCAL       ;PERFORM SEEK TO 0
 4178  015710  010065  000002                MOV     R0,P.CYLN(R5)   ;SET CURRENT CYLINDER NUMBER
 4179  015714  112765  000123  000001        MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
 4180  015722  004737  030146                JSR     PC,DRVCAL       ;WRITE A SECTOR
 4181  015726  112737  000003  003143        MOVB    #3,TSTTYP       ;SET INDICATOR FOR SELF-TEST
 4182  015734  004737  017300                JSR     PC,REDOFS       ;READ SECTOR WITH RANGE OF OFFSETS
 4183  015740  105265  000005                INCB    P.TRCK(R5)      ;INCREMENT TRACK
 4184  015744  126527  000005  000002        CMPB    P.TRCK(R5),#2   ;SEE IF ALL TRACKS DONE ON THIS CYL
 4185  015752  101745                        BLOS    6$              ;BR IF NOT YET
 4186  015754  004737  017644                JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUMS
 4187  015760  062765  000005  000004        ADD     #5,P.SECT(R5)   ;INCREMENT SECTOR BY 5
 4188  015766  126527  000004  000023        CMPB    P.SECT(R5),#23  ;SEE IF ALL SECTORS DONE ON THIS TRACK
 4189  015774  101716                        BLOS    4$              ;BR IF NOT YET
 4190  015776  063700  024412                ADD     HOLD3,R0        ;INCR CYL NO. BY 100/200
 4191  016002  005303                        DEC     R3              ;DECR CYL CNTR
 4192  016004  001307                        BNE     3$              ;BR IF NOT DONE WITH ALL CYLS YET
 4193  016006  004737  020024                JSR     PC,AVSCOR       ;COMPUTE AVG SELF-TEST SCORES
 4194  016012  105037  003143                CLRB    TSTTYP          ;CLEAR OVRWRT TEST INDICATOR
 4195
 4196
 4197
 4198                           ;*****************************************************************
 4199                           ;*TEST 10       COMPATIBILITY DATA READ TEST
 4200                           ;*
 4201                           ;*      HAVING ESTABLISHED A SELF-TEST SCORE FOR THIS DRIVE, THE
 4202                           ;*      PROGRAM PROCEEDS TO PERFORM THE COMPATIBILITY DATA READS
 4203                           ;*      OF THE PATTERNS WRITTEN BY ALL THE DRIVES IN EACH CYLINDER
 4204                           ;*      BLOCK (ON EACH SURFACE). EACH COMPATIBILITY CYLINDER BLOCK
 4205                           ;*      SHOWN IN TABLE C IS READ  WITH THE FOLLOWING OFFSET VALUES :
 4206                           ;*      100,200,300,400,500,600,700,800,900,1000,1100,1200 MICRO-IN.,
 4207                           ;*      IN (+) AND (-) DIRECTIONS. THE PROGRAM SCANS FOR READ ERRORS
 4208                           ;*      DURING EACH READ, AND IF ONE OCCURS, THE PROGRAM DETERMINES
 4209                           ;*      WHICH DRIVE'S DATA WAS BEING READ AT THAT INSTANT AND A SCORE
 4210                           ;*      FOR THAT DRIVE IS DECREMENTED. THEN, THE TRANSFER IS CONTINUED
 4211                           ;*      AT THE NEXT SECTOR, WITH THAT OFFSET VALUE. THE READS ARE
 4212                           ;*      DONE WITH ALL OF THE OFFSETS APPLIED, AND A SEPARATE SCORE
 4213                           ;*      FOR EACH DRIVE IS KEPT, WHILE THE CURRENT DRIVE IS READING
 4214                           ;*      THE COMPATIBILITY DATA. THEN, EACH SCORE IS APPROPRIATELY
 4215                           ;*      ADJUSTED TO REFLECT THE SELF-TEST SCORE FOR THE CURRENT DRIVE
 4216                           ;*      ON THAT SURFACE. THE SCORES ARE THEN AVERAGED OVER ALL CYLINDER
 4217                           ;*      BLOCKS. EACH SCORE IS PROPORTIONAL TO THE  CAPABILITY OF THE
 4218                           ;*      CURRENT DRIVE TO SUCCESSFULLY READ THE DATA WRITTEN BY ONE OF
 4219                           ;*      THE OTHER DRIVES, AND SCORES ARE COMPUTED SEPARATELY FOR EACH
 4220                           ;*      SURFACE (TRACK). THUS, THE SCORES REVEAL WHICH DRIVES ARE
 4221                           ;*      INVOLVED IN A SITUATION IN WHICH A PARTICULAR DRIVE HAS
 4222                           ;*      DIFFICULTY IN READING THE DATA OF ONE OR SEVERAL OTHER DRIVES.
 4223                           ;*
 4224                           ;*****************************************************************
```

```
4225  016016  000004                    TST10:  SCOPE
4226  016020  012737  000010  001334            MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4227  016026  004737  022042                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4228  016032  004737  021170                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4229  016036  004737  020774                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4230  016042  112737  000002  003143            MOVB    #2,TSTTYP       ;SET INDIC FOR READ TEST
4231  016050  105037  003147                    CLRB    DIF100          ;CLEAR "CYL DIF = 100/200" FLAG
4232  016054  012700  007112                    MOV     #NCSCRO,R0      ;GET ADRS OF COMPAT READ SCORES
4233  016060  005020                    6$:     CLR     (R0)+           ;INIT A SCORE TO 0
4234  016062  020027  007412                    CMP     R0,#NCSCRO+192. ;SEE IF ALL SCORES CLEARED YET
4235  016066  103774                            BLO     6$              ;BR IF NOT YET
4236  016070  012703  000006                    MOV     #6,R3           ;SET CYL BLK CNTR = 6
4237  016074  004737  022642                    JSR     PC,INTBLD       ;INIT TABLE D
4238  016100  013701  024414                    MOV     HOLD4,R1        ;SET START CYL = 60/160
4239  016104  012765  053342  000010            MOV     #RWBUF,P.BAL0(R5) ;SET BUS ADRS
4240  016112  052765  100000  000014            BIS     #DTBAII,P.PRST(R5) ;SET BUS ADRS INCR INHIBIT
4241  016120  012765  137000  000012            MOV     #-16896.,P.WC(R5) ;SET 22-SECTOR WORD COUNT
4242  016126  105737  003134                    TSTB    FORMAT          ;TEST THE FORMAT
4243  016132  001403                            BEQ     10$             ;BR IF 22 SECTORS
4244  016134  012765  142000  000012            MOV     #-15360.,P.WC(R5) ;SET 20-SECTOR WORD COUNT
4245  016142  012700  000020            10$:    MOV     #16.,R0         ;INIT CYL CNTR = 16.
4246  016146  010165  000002                    MOV     R1,P.CYLN(R5)   ;SET CYL NO.
4247  016152  004737  017300            12$:    JSR     PC,REDOFS       ;DO READS WITH OFSTS ON THIS CYL
4248  016156  004737  017644                    JSR     PC,ADSCOR       ;ADD SCRATCH SCORES TO SUM
4249  016162  005265  000002                    INC     P.CYLN(R5)      ;INCR CYL NO.
4250  016166  005300                            DEC     R0              ;SEE IF DONE WITH THIS CYL BLK
4251  016170  001370                            BNE     12$             ;BR IF NOT YET
4252  016172  004737  023022                    JSR     PC,ROTBLD       ;ROTATE TABLED FOR NEXT CURRENT ZONE
4253  016176  063701  024412                    ADD     HOLD3,R1        ;INCR CYL BY 100/200
4254  016202  112737  000001  003147            MOVB    #1,DIF100       ;SET "CYL DIF = 100" FLAG
4255  016210  005303                            DEC     R3              ;SEE IF DONE WITH ALL CYL BLKS
4256  016212  001353                            BNE     10$             ;BR IF NOT YET
4257  016214  004737  020024                    JSR     PC,AVSCOR       ;COMPUTE AVERAGES OF READ SCORES
4258  016220  105037  003143                    CLRB    TSTTYP          ;CLEAR OFST TEST INDICATOR
4259                            ;ADJUST COMPAT READ SCORES (- OFSTS) BY SELF-TEST SCORES
4260  016224  012701  007112                    MOV     #NCSCRO,R1      ;SET COMPAT SCORE POINTER
4261  016230  012703  007412                    MOV     #NSSCRO,R3      ;SET SELF-TEST SCORE POINTER
4262  016234  012700  005512            14$:    MOV     #DRVLST,R0      ;SET DRIVE LIST  POINTER
4263  016240  012737  000014  001304            MOV     #12.,$TMP11     ;COMPUTE DIFFERENCE OF 12 (DEC) AND
4264                                            ;     SELF-TEST SCORE FOR THIS TRACK
4265  016246  162337  001304                    SUB     (R3)+,$TMP11
4266  016252  010104                            MOV     R1,R4           ;GET READ SCORE POINTER IN R4
4267  016254  005720                    15$:    TST     (R0)+           ;SEE IF A DRIVE LISTED HERE
4268  016256  001417                            BEQ     18$             ;BR IF NOT
4269  016260  013737  001304  001306            MOV     $TMP11,$TMP12   ;PUT DIFFERENCE INTO $TMP12
4270  016266  061437  001306                    ADD     (R4),$TMP12     ;ADD DIFF TO READ SCORE
4271  016272  023727  001306  000014            CMP     $TMP12,#12.     ;SEE IF 12 EXCEEDED
4272  016300  101403                            BLOS    16$             ;BR IF NOT
4273  016302  012737  000014  001306            MOV     #12.,$TMP12     ;SET SCORE = 12.
4274  016310  013724  001306            16$:    MOV     $TMP12,(R4)+    ;PUT BACK ADJUSTED SCORE
4275  016314  000757                            BR      15$             ;BR TO CONTINUE
4276  016316  062701  000040            18$:    ADD     #32.,R1         ;POINT TO SCORES FOR NEXT TRACK
4277  016322  020127  007212                    CMP     R1,#NCSCRO+64.  ;SEE IF SCORE TABLE EXCEEDED
4278  016326  101742                            BLOS    14$             ;BR IF NOT YET
4279                            ;ADJUST COMPAT READ SCORES (+ OFSTS) BY SELF-TEST SCORES
4280  016330  012701  007252                    MOV     #PCSCRO,R1      ;SET COMPAT SCORE POINTER
```

```
4281  016334  012703  007420                  MOV     #PSSCRO,R3      ;SET SELF-TEST SCORE POINTER
4282  016340  012700  005512          24$:    MOV     #DRVLST,R0      ;SET DRIVE LIST  POINTER
4283  016344  012737  000014  001304          MOV     #12.,$TMP11     ;COMPUTE DIFFERENCE OF 12 (DEC) AND
4284                                                                  ;     SELF-TEST SCORE FOR THIS TRACK
4285  016352  162337  001304                  SUB     (R3)+,$TMP11
4286  016356  010104                          MOV     R1,R4           ;GET READ SCORE POINTER IN R4
4287  016360  005720                  25$:    TST     (R0)+           ;SEE IF A DRIVE LISTED HERE
4288  016362  001417                          BEQ     28$             ;BR IF NOT
4289  016364  013737  001304  001306          MOV     $TMP11,$TMP12   ;PUT DIFFERENCE INTO $TMP12
4290  016372  061437  001306                  ADD     (R4),$TMP12     ;ADD DIFF TO READ SCORE
4291  016376  023727  001306  000014          CMP     $TMP12,#12.     ;SEE IF 12 EXCEEDED
4292  016404  101403                          BLOS    26$             ;BR IF NOT
4293  016406  012737  000014  001306          MOV     #12.,$TMP12     ;SET SCORE = 12.
4294  016414  013724  001306          26$:    MOV     $TMP12,(R4)+    ;PUT BACK ADJUSTED SCORE
4295  016420  000757                          BR      25$             ;BR TO CONTINUE
4296  016422  062701  000040          28$:    ADD     #32.,R1         ;POINT TO SCORES FOR NEXT TRACK
4297  016426  020127  007352                  CMP     R1,#PCSCRO+64.  ;SEE IF SCORE TABLE EXCEEDED
4298  016432  101742                          BLOS    24$             ;BR IF NOT YET
4299
4300
4301
4302                                  ;;*****************************************************************
4303                                  ;*TEST 11       TYPE TEST SCORES, DISMOUNT PACK, IN PASS 2
4304                                  ;*
4305                                  ;*      THE OVERWRITE AND COMPATIBILITY DATA READ TEST SCORES FOR THIS
4306                                  ;*      DRIVE ARE CONVERTED AND TYPED. THEN, THE OPERATOR IS
4307                                  ;*      DIRECTED TO UNLOAD THE DRIVE CURRENTLY UNDER TEST, AND TO REMOVE
4308                                  ;*      THE TEST PACK. THEN, THE PROGRAM DOES 1 OF 4 THINGS :
4309                                  ;*      (1) IF THERE IS ANOTHER DRIVE TO TEST ON THIS SUBSYS, THE
4310                                  ;*      PROGRAM JUMPS TO TEST 5 FOR THE NEXT DRIVE.
4311                                  ;*      (2) IF THERE IS ONLY ONE SUBSYS (FROM 200 START), AND IF
4312                                  ;*      PASS 2 HAS BEEN COMPLETED ON ALL DRIVES, THE ENTIRE TESTING AND
4313                                  ;*      REPORTING HAVE BEEN COMPLETED, AND THE PROGRAM TYPES A
4314                                  ;*      COMPLETION MESSAGE, AND THEN IT RESTARTS AT ADRS 200.
4315                                  ;*      (3) IF THERE IS ANOTHER SUBSYS TO PERFORM PASS 2 ON, THE
4316                                  ;*      PROGRAM TELLS THE OPERATOR TO RESTART AT ADDRESS 220, AND THEN
4317                                  ;*      IT HALTS.
4318                                  ;*      (4) IF THERE ARE NO MORE DRIVES TO TEST ON ANY OTHER SUBSYS, THE
4319                                  ;*      PROGRAM TYPES A COMPLETION MESSAGE AND THEN IT HALTS.
4320                                  ;*
4321                                  ;*****************************************************************
4322  016434  000004                  TST11:  SCOPE
4323  016436  012737  000011  001334          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
4324  016444  004737  022042                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
4325  016450  004737  021170                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
4326  016454  004737  020774                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
4327  016460  112737  000001  003127          MOVB    #1,TSTING       ;ALLOW TTY ESCAPE
4328                                  ;CONVERT AND TYPE TEST SCORES
4329  016466  013700  005554                  MOV     DRVPTR,R0       ;GET DRIVE LIST POINTER
4330  016472  112037  010743                  MOVB    (R0)+,RESLTS+20. ;GET NO. OF THIS DRIVE
4331  016476  111037  010742                  MOVB    (R0),RESLTS+19.
4332  016502  104401  010717                  TYPE    ,RESLTS         ;TYPE TEST RESULT HEADINGS
4333  016506  012737  006612  001306          MOV     #NOSCRO,$TMP12  ;GET POINTER TO OVRWRT SCORES
4334  016514  012700  000060                  MOV     #'0,R0          ;INIT TRACK NO. TO 0
4335  016520  012701  005512          5$:     MOV     #DRVLST,R1      ;GET DRIVE LIST POINTER
4336  016524  110037  011225                  MOVB    R0,TRKBUF+1     ;SET TRACK NO. FOR PRINTOUT
```

```
 4337  016530  013704  001306            MOV     $TMP12,R4       ;GET SCORE POINTER
 4338  016534  004737  025564      6$:   JSR     PC,CTLOUT       ;CHECK FOR TTY ESCAPE RQST
 4339  016540  104401  011224            TYPE    ,TRKBUF         ;TYPE TRACK (HEAD) NUMBER
 4340  016544  012103                    MOV     (R1)+,R3        ;GET A DRIVE NAME
 4341  016546  020377  167002            CMP     R3,@DRVPTR      ;SEE IF IT IS CURRENT DRIVE
 4342  016552  001003                    BNE     8$              ;BR IF NOT
 4343  016554  104401  011235            TYPE    ,SELF           ;TYPE "SELF"
 4344  016560  000407                    BR      10$             ;CONTINUE
 4345  016562  110337  011232      8$:   MOVB    R3,DRVBUF+2     ;GET DRIVE NO.
 4346  016566  000303                    SWAB    R3
 4347  016570  110337  011231            MOVB    R3,DRVBUF+1     ;GET SUBSYS NAME
 4348  016574  104401  011230            TYPE    ,DRVBUF         ;TYPE NO. OF DRIVE READ
 4349  016600  005037  005212      10$:  CLR     BUFFO           ;CLEAR TTY CHAR BUF
 4350  016604  005037  005214            CLR     BUFFO+2
 4351  016610  010437  001310            MOV     R4,$TMP13       ;GET SCORE POINTER
 4352  016614  012437  005212            MOV     (R4)+,BUFFO     ;PUT AN OVRWRT SCORE INTO BUF
 4353  016620  012737  000004  001304    MOV     #4,$TMP11       ;SET COUNTER = 4
 4354  016626  023727  005212  000012 11$:  CMP   BUFFO,#10.      ;SEE IF SCORE < 10.
 4355  016634  103002                    BHIS    12$             ;BR IF NOT
 4356  016636  104401  012150            TYPE    ,SPACE1         ;TYPE 1 SPACE
 4357  016642  023727  005212  000007 12$:  CMP   BUFFO,#7        ;SEE IF SCORE > 7
 4358  016650  101003                    BHI     14$             ;BR IF YES
 4359  016652  104401  012170            TYPE    ,PROMPT         ;TYPE "*  "
 4360  016656  000402                    BR      16$             ;CONTINUE
 4361  016660  104401  012147      14$:  TYPE    ,SPACE2         ;TYPE 2 SPACES
 4362  016664  012746  005212      16$:  MOV     #BUFFO,-(SP)    ;GET BUF ADRS ON STACK
 4363  016670  004737  043174            JSR     PC,@#$DB2D      ;CONVERT SCORE TO DEC.
 4364  016674  004737  043370            JSR     PC,@#$SUPRS     ;TYPE IT
 4365  016700  104401  011243            TYPE    ,TAB            ;TYPE A TAB
 4366  016704  062737  000140  001310    ADD     #96.,$TMP13
 4367  016712  017737  162372  005212    MOV     @$TMP13,BUFFO   ;PUT NEXT SCORE INTO BUFFER
 4368  016720  005337  001304            DEC     $TMP11          ;DECREMENT SCORE COUNTER
 4369  016724  001340                    BNE     11$             ;BR IF READ SCORE SHOULD BE TYPED
 4370  016726  104401  001325            TYPE    ,$CRLF          ;TYPE <CR>, <LR>
 4371  016732  005711                    TST     (R1)            ;SEE IF ALL SCORES TYPED FOR THIS TRACK
 4372  016734  001277                    BNE     6$              ;BR IF NOT
 4373  016736  104401  001325            TYPE    ,$CRLF          ;TYPE <CR>, <LF>
 4374  016742  005200                    INC     R0              ;INCR TRACK NO.
 4375  016744  062737  000040  001306    ADD     #32.,$TMP12     ;INCREMENT SCORE POINTER FOR NEXT TRACK
 4376  016752  020027  000062            CMP     R0,#'2          ;SEE IF ALL TRACKS TYPED YET
 4377  016756  101660                    BLOS    5$              ;BR IF NOT YET
 4378                          ;COMPUTE, CONVERT, AND TYPE TEST AVERAGES (OVER ALL SURFACES)
 4379  016760  004737  020262            JSR     PC,TRKAVG       ;GET AVGS OVER ALL TRKS
 4380  016764  013700  005554            MOV     DRVPTR,R0       ;PUT DRIVE NAME INTO BUF
 4381  016770  112037  011312            MOVB    (R0)+,BADDRV+7
 4382  016774  111037  011311            MOVB    (R0),BADDRV+6
 4383  017000  005037  001304            CLR     $TMP11          ;CLEAR SCORE INDICATOR
 4384  017004  013737  007426  005212    MOV     OVRAVG,BUFFO    ;GET OVRWRT AVG SCORE
 4385  017012  012737  011132  017026    MOV     #OAVERG,22$
 4386  017020  104401  011303      20$:  TYPE    ,BADDRV         ;TYPE "DRIVE XX"
 4387  017024  104401                    TYPE                    ;TYPE WHICH KIND OF AVERAGE
 4388  017026  000000      22$:  .WORD   0
 4389  017030  012746  005212            MOV     #BUFFO,-(SP)    ;PUT POINTER TO AVG ON STACK
 4390  017034  004737  043174            JSR     PC,@#$DB2D      ;CONVERT TO DECIMAL
 4391  017040  004737  043370            JSR     PC,@#$SUPRS     ;TYPE IT
 4392  017044  104401  001325            TYPE    ,$CRLF          ;TYPE <CR>, <LF>
```

```
 4393  017050  013737  007430  005212          MOV     COMAVG,BUFFO    ;GET READ AVG SCORE
 4394  017056  012737  011151  017026          MOV     #RAVERG,22$
 4395  017064  005137  001304                  COM     $TMP11          ;COMPLEMENT SCORE INDIC
 4396  017070  001353                          BNE     20$             ;BR TO TYPE READ SCORE
 4397  017072  104401  001325                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 4398                                ;DISMOUNT PACK ON THIS DRIVE
 4399  017076  105037  003127                  CLRB    TSTING          ;DON'T ALLOW TTY ESCAPE
 4400  017102  004737  022472                  JSR     PC,DMCART       ;DIRECT OPERATOR TO DISMOUNT PACK
 4401  017106  062737  000002  005554          ADD     #2,DRVPTR       ;INCREMENT DRIVE LIST POINTER
 4402  017114  013700  005554                  MOV     DRVPTR,R0
 4403  017120  005710                          TST     (R0)            ;SEE IF ALL DONE WITH PASS 2 YET
 4404  017122  001421                          BEQ     28$             ;BR IF DONE WITH PASS 2
 4405  017124  126037  000001  005510          CMPB    1(R0),SUBSYS    ;SEE IF DONE WITH THIS SUBSYS YET
 4406  017132  001005                          BNE     24$             ;BR IF DONE WITH THIS SUBSYS
 4407  017134  112737  000004  001102          MOVB    #4,$TSTNM       ;SET TEST NO. = 4
 4408  017142  000137  015146                  JMP     TST5            ;DO PASS 2 ON NEXT DRIVE OF SUBSYS
 4409  017146  116037  000001  010574  24$:     MOVB    1(R0),STR220+39. ;PUT SUBSYS NAME INTO MSG
 4410  017154  104401  010525                  TYPE    ,STR220         ;TYPE RESTRT MSG FOR PASS 2
 4411  017160  000000                  26$:     HALT                    ;HALT THE PROCESSOR SO THAT OPERATOR
 4412  017162  000137  000220                  JMP     220             ;  CAN RESTART AT PROPER ADRS
 4413  017166  104401  011170          28$:     TYPE    ,ENDTST         ;TYPE "**  END OF TESTING  **"
 4414  017172  012737  005512  005554          MOV     #DRVLST,DRVPTR  ;RE-INIT DRIVE LIST POINTER
 4415  017200  105737  003126                  TSTB    MDFLAG          ;SEE IF 200 START
 4416  017204  001002                          BNE     30$             ;BR IF NOT
 4417  017206  000137  000200                  JMP     200             ;RESTART AT ADRS 200
 4418  017212  000000                  30$:     HALT                    ;HALT THE PROCESSOR
 4419  017214  000137  000204                  JMP     204             ;RESTART AT ADRS 204
 4420
 4421
 4422
 4423                                .SBTTL  END OF PASS ROUTINE
 4424
 4425                                ;;**************************************************************
 4426                                ;*INCREMENT THE PASS NUMBER ($PASS)
 4427                                ;*IF THERES A MONITOR GO TO IT
 4428                                ;*IF THERE ISN'T JUMP TO 200
 4429
 4430  017220                        $EOP:
 4431  017220  000004                          SCOPE
 4432  017222  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 4433  017226  005237  001336                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 4434  017232  042737  100000  001336          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 4435  017240  005327                          DEC     (PC)+           ;;LOOP?
 4436  017242  000001                  $EOPCT: .WORD   1
 4437  017244  003013                          BGT     $DOAGN          ;;YES
 4438  017246  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 4439  017250  000001                  $ENDCT: .WORD   1
 4440  017252  017242                          $EOPCT
 4441  017254  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 4442  017260  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 4443  017262  000005                          RESET                   ;;CLEAR THE WORLD
 4444  017264  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 4445  017266  000240                          NOP                     ;;SAVE ROOM
 4446  017270  000240                          NOP                     ;;FOR
 4447  017272  000240                          NOP                     ;;ACT11
 4448  017274                        $DOAGN:
```

```
 4449  017274  000137  000200                    JMP     @#200           ;;RETURN
 4450
 4451
 4452
 4453                                    ;;*****************************************************************
 4454                                    ;*REDOFS - READ DATA WITH RANGE OF OFFSETS
 4455                                    ;*THIS SUBROUTINE PERFORMS READS WITH OFFSETS LISTED IN OFSTBL.
 4456                                    ;*THE OFFSETS ARE APPLIED MINIMUM TO MAXIMUM, IN BOTH DIRECTIONS,
 4457                                    ;*UNTIL THERE IS NO FAILURE. EACH TIME A READ ERROR OCCURS,
 4458                                    ;*THE APPROPRIATE SCORE FOR THE DRIVE INVOLVED IS INCREMENTED NO FURTHER
 4459                                    ;*(IN THE SCRATCH SCORE TABLE) . EACH READ IS DONE TO A PACK
 4460                                    ;*AREA, WHOSE ADDRESS IS IN THE PARAMETER BLOCK ON ENTRY.
 4461                                    ;*****************************************************************
 4462  017300  104407                    REDOFS: SAVREG                  ;SAVE R0-R5
 4463                                    ;INITIALIZE SCRATCH SCORES
 4464  017302  122737  000003  003143            CMPB    #3,TSTTYP       ;SEE IF DRIVE SELF TEST
 4465  017310  001406                            BEQ     5$              ;BR IF YES
 4466  017312  012700  007432                    MOV     #NSCORO,R0      ;GET ADRS OF SCRATCH SCORE TABLE
 4467  017316  105020                    4$:     CLRB    (R0)+           ;INIT A SCORE TO 0
 4468  017320  020027  007572                    CMP     R0,#NSCORO+96.  ;SEE IF DONE YET
 4469  017324  103774                            BLO     4$              ;BR IF NOT YET
 4470                                    ;PERFORM READS WITH OFFSET
 4471  017326  112765  000117  000001    5$:     MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 4472  017334  105737  003147                    TSTB    DIF100          ;SEE IF CYL DIF = 100/200 (OCT)
 4473  017340  001412                            BEQ     9$              ;BR IF NOT
 4474  017342  163765  024416  000002            SUB     HOLD5,P.CYLN(R5)        ;SEEK HALF THE CYLS
 4475  017350  004737  030146                    JSR     PC,DRVCAL
 4476  017354  063765  024416  000002            ADD     HOLD5,P.CYLN(R5)        ;SEEK THE REST OF THE WAY
 4477  017362  105037  003147                    CLRB    DIF100          ;CLEAR "CYL DIF = 100/200 FLAG"
 4478  017366  004737  030146          9$:       JSR     PC,DRVCAL       ;PERFORM SEEK
 4479  017372  105037  003156                    CLRB    OFSDIR          ;CLEAR OFFSET DIRECTION FLAG
 4480  017376  012703  000014                    MOV     #12.,R3         ;SET OFFSET COUNTER FOR NEG OFSTS
 4481  017402  012700  000204                    MOV     #204,R0         ;INIT NEG OFST VALUE TO 100 UIN
 4482  017406  105037  003157          12$:      CLRB    OFSCNT          ;INIT OFFSET NUMBER
 4483  017412  122737  000003  003143  6$:       CMPB    #3,TSTTYP       ;SEE IF SELF-TEST
 4484  017420  001020                            BNE     15$             ;BR IF NOT
 4485  017422  116501  000005                    MOVB    P.TRCK(R5),R1   ;GET TRACK NUMBER
 4486  017426  006301                            ASL     R1              ;MULT BY 16.
 4487  017430  006301                            ASL     R1
 4488  017432  006301                            ASL     R1
 4489  017434  006301                            ASL     R1
 4490  017436  062701  007432                    ADD     #NSCORO,R1      ;GET POINTER TO SCORES
 4491  017442  105737  003156                    TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
 4492  017446  001402                            BEQ     17$             ;BR IF NEGATIVE
 4493  017450  062701  000060                    ADD     #48.,R1         ;POINT TO POSITIVE SCORES
 4494  017454  012704  000001          17$:      MOV     #1,R4           ;SET COUNTER TO 1
 4495  017460  000411                            BR      14$             ;CONTINUE
 4496  017462  012701  007432          15$:      MOV     #NSCORO,R1      ;GET POINTER TO SCORES
 4497  017466  012704  000060                    MOV     #48.,R4         ;SET COUNTER TO 48.
 4498  017472  105737  003156                    TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
 4499  017476  001402                            BEQ     14$             ;BR IF NEGATIVE
 4500  017500  012701  007512                    MOV     #PSCORO,R1      ;GET POINTER FOR POS SCORES
 4501  017504  122137  003157          14$:      CMPB    (R1)+,OFSCNT    ;SEE IF THIS IS A PERFECT SCORE YET
 4502  017510  001002                            BNE     18$             ;BR IF NOT
 4503  017512  105261  177777                    INCB    -1(R1)          ;INCREMENT THIS SCORE
 4504  017516  005304                    18$:    DEC     R4              ;DECREMENT COUNTER
```

```
 4505  017520  001371                        BNE     14$                     ;BR IF NOT DONE YET
 4506  017522  110065  000006          16$:  MOVB    R0,P.OFST(R5) ;GET AN OFFSET VALUE
 4507  017526  112765  000115  000001        MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET CMND
 4508  017534  004737  030146                JSR     PC,DRVCAL               ;PERFORM OFFSET
 4509  017540  105237  003157                INCB    OFSCNT                  ;INCR OFFSET NUMBER
 4510  017544  112765  000121  000001        MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 4511  017552  004737  027500                JSR     PC,SVPRMS               ;SAVE PARAMETERS OF TRANSFER
 4512  017556  004737  027570                JSR     PC,TRNSFR               ;DO XFER, HANDLE DCK ERRORS
 4513  017562  112765  000177  000001        MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR COMMAND
 4514  017570  004737  030146                JSR     PC,DRVCAL               ;DO A SUBSYSTEM CLEAR
 4515  017574  105065  000006                CLRB    P.OFST(R5)              ;CLEAR OFFSET VALUE
 4516  017600  112765  000115  000001        MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
 4517  017606  004737  030146                JSR     PC,DRVCAL               ;PERFORM OFFSET TO 0
 4518  017612  062700  000004                ADD     #4,R0                   ;INCR OFST VALUE BY 100 UIN
 4519  017616  005303                        DEC     R3                      ;DECR COUNTER
 4520  017620  001274                        BNE     6$                      ;BR IF NOT DONE IN THIS DIRECTION YET
 4521  017622  012703  000014                MOV     #12.,R3                 ;SET COUNTER FOR POS SFSTS
 4522  017626  012700  000004                MOV     #4,R0                   ;INIT POS OFST TO 100 UIN
 4523  017632  105137  003156                COMB    OFSDIR                  ;COMPLEMENT OFFSET DIRECTION FLAG
 4524  017636  001263                        BNE     12$                     ;BR IF POS OFSTS NOT DONE YET
 4525  017640  104410                        RESREG                          ;RESTORE R0-R5
 4526  017642  000207                        RTS     PC                      ;RETURN
 4527
 4528
 4529
 4530                                  ;;*****************************************************************
 4531                                  ;*ADSCOR - ADD THE + AND - SCRATCH SCORES TO THE + AND - SCORE
 4532                                  ;*SUMS, FOR EITHER THE OVRWRT, SELF,  OR COMPAT READ TEST, AS DETERMINED
 4533                                  ;*BY "TSTTYP".
 4534                                  ;;*****************************************************************
 4535  017644  104407                  ADSCOR: SAVREG                          ;SAVE R0-R5
 4536  017646  122737  000003  003143          CMPB    #3,TSTTYP               ;SEE IF SELF-TEST
 4537  017654  001031                        BNE     2$                      ;BR IF NOT
 4538  017656  113700  007432                MOVB    NSCOR0,R0               ;ADD SELF-TEST SCORES
 4539  017662  060037  007412                ADD     R0,NSSCR0
 4540  017666  113700  007452                MOVB    NSCOR1,R0
 4541  017672  060037  007414                ADD     R0,NSSCR1
 4542  017676  113700  007472                MOVB    NSCOR2,R0
 4543  017702  060037  007416                ADD     R0,NSSCR2
 4544  017706  113700  007512                MOVB    PSCOR0,R0
 4545  017712  060037  007420                ADD     R0,PSSCR0
 4546  017716  113700  007532                MOVB    PSCOR1,R0
 4547  017722  060037  007422                ADD     R0,PSSCR1
 4548  017726  113700  007552                MOVB    PSCOR2,R0
 4549  017732  060037  007424                ADD     R0,PSSCR2
 4550  017736  000430                        BR      8$                      ;EXIT
 4551  017740  012700  006612          2$:   MOV     #NOSCR0,R0              ;GET POINTER TO (-) OVRWRT SCORES
 4552  017744  012705  006752                MOV     #POSCR0,R5              ;GET POINTER TO (+) OVRWRT SCORES
 4553  017750  122737  000001  003143        CMPB    #1,TSTTYP               ;SEE IF READING OVRWRT DATA
 4554  017756  001404                        BEQ     4$                      ;BR IF YES
 4555  017760  012700  007112                MOV     #NCSCR0,R0              ;GET POINTER TO (-) COMPAT DATA SCORES
 4556  017764  012705  007252                MOV     #PCSCR0,R5              ;GET POINTER TO (+) COMPAT DATA SCORES
 4557  017770  012704  000060          4$:   MOV     #48.,R4                 ;INIT COUNTER TO 48.
 4558  017774  012701  007432                MOV     #NSCOR0,R1              ;GET POINTER TO NEG SCRATCH SCORES
 4559  020000  012703  007512                MOV     #PSCOR0,R3              ;GET POINTER TO POS SCRATCH SCORES
 4560  020004  112102                  6$:   MOVB    (R1)+,R2                ;GET A SCRATCH NEG SCORE
```

```
 4561  020006  060220                          ADD     R2,(R0)+        ;ADD IT TO (-) SCORE SUM
 4562  020010  112302                          MOVB    (R3)+,R2        ;GET PNTR TO POS SCRATCH SCORES
 4563  020012  060225                          ADD     R2,(R5)+        ;ADD IT TO (+) SCORE SUM
 4564  020014  005304                          DEC     R4              ;DECREMENT COUNTER
 4565  020016  001372                          BNE     6$              ;BR IF NOT DONE YET
 4566  020020  104410                  8$:     RESREG                  ;RESTORE R0-R5
 4567  020022  000207                          RTS     PC              ;RETURN
 4568
 4569
 4570
 4571                                  ;;*****************************************************************
 4572                                  ;*AVSCOR - COMPUTE THE + AND - AVERAGE SCORES FOR THE OVRWRT, COMPATIBILITY
 4573                                  ;*DATA TEST, OR SELF-TEST, AS DETERMINED BY "TSTTYP". THE SUMS IN THE APPROPRIATE
 4574                                  ;*SCORE TABLES ARE EACH DIVIDED BY THE NUMBER OF CYLINDERS READ, TO
 4575                                  ;*COMPUTE THE AVERAGES.
 4576                                  ;;*****************************************************************
 4577  020024  104407                  AVSCOR: SAVREG                  ;SAVE R0-R5
 4578  020026  122737  000003  003143          CMPB    #3,TSTTYP       ;SEE IF SELF-TEST
 4579  020034  001014                          BNE     2$              ;BR IF NOT
 4580  020036  012737  007412  001304          MOV     #NSSCRO,$TMP11  ;GET ADRS OF (-) SELF-TEST SCORES
 4581  020044  012737  007420  001310          MOV     #PSSCRO,$TMP13  ;GET ADRS OF (+) SELF-TEST SCORES
 4582  020052  012705  000030                  MOV     #24.,R5         ;SET DIVISOR = 24.
 4583  020056  012737  000003  001306          MOV     #3,$TMP12       ;SET COUNTER = 3
 4584  020064  000427                          BR      12$             ;CONTINUE
 4585  020066  012737  006612  001304  2$:     MOV     #NOSCRO,$TMP11  ;GET POINTER TO (-) OVRWRT SCORES
 4586  020074  012737  006752  001310          MOV     #POSCRO,$TMP13  ;GET POINTER TO (+) OVRWRT SCORES
 4587  020102  012705  000007                  MOV     #7.,R5          ;SET DIVISOR = 7.
 4588  020106  122737  000001  003143          CMPB    #1,TSTTYP       ;SEE IF READING OVRWRT DATA
 4589  020114  001410                          BEQ     4$              ;BR IF YES
 4590  020116  012737  007112  001304          MOV     #NCSCRO,$TMP11  ;GET POINTER TO (-) COMPAT DATA SCORES
 4591  020124  012737  007252  001310          MOV     #PCSCRO,$TMP13  ;GET POINTER TO (+) COMPAT DATA SCORES
 4592  020132  012705  000140                  MOV     #96.,R5         ;SET DIVISOR = 96.
 4593  020136  012737  000060  001306  4$:     MOV     #48.,$TMP12     ;SET COUNTER = 48.
 4594  020144  105037  003156          12$:    CLRB    OFSDIR          ;INIT FLAG FOR (-) DIRECTION
 4595  020150  005000                  6$:     CLR     R0              ;CLEAR HI BITS OF DIVIDEND AND DIVISOR
 4596  020152  005001                          CLR     R1
 4597  020154  005002                          CLR     R2
 4598  020156  005004                          CLR     R4
 4599  020160  017703  161120                  MOV     @$TMP11,R3      ;SET DIVIDEND = (-) SUM
 4600  020164  105737  003156                  TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
 4601  020170  001402                          BEQ     10$
 4602  020172  017703  161112                  MOV     @$TMP13,R3      ;SET DIVIDEND = (+) SUM
 4603  020176  004737  042676          10$:    JSR     PC,M.DPID       ;DIVIDE TO GET AVERAGES
 4604  020202  010500                          MOV     R5,R0           ;GET DIVISOR
 4605  020204  006200                          ASR     R0              ;DIVIDE IT BY 2
 4606  020206  020100                          CMP     R1,R0           ;SEE IF REMNDR > HALF DIVISOR
 4607  020210  103401                          BLO     8$              ;BR IF NOT
 4608  020212  005203                          INC     R3              ;ADD 1 TO QUOTIENT
 4609  020214  105137  003156          8$:     COMB    OFSDIR          ;COMPL DIR FLAG
 4610  020220  001406                          BEQ     14$             ;BR FOR (+)
 4611  020222  010377  161056                  MOV     R3,@$TMP11      ;PUT (-) AVG INTO TABLE
 4612  020226  062737  000002  001304          ADD     #2,$TMP11
 4613  020234  000745                          BR      6$
 4614  020236  010377  161046          14$:    MOV     R3,@$TMP13      ;PUT (+) AVG INTO TABLE
 4615  020242  062737  000002  001310          ADD     #2,$TMP13
 4616  020250  005337  001306                  DEC     $TMP12          ;DECREMENT COUNTER
```

```
                                        BNE     12$             ;BR IF NOT DONE YET
                                        RESREG                  ;RESTORE R0-R5
                                        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.
                                ;;*****************************************************************
020262  104407                  TRKAVG: SAVREG                  ;SAVE R0-R5
                                ;TAKE SUMS OF OVERWRITE AND COMPAT READ TEST SCORES OVER ALL TRACKS
020264  012700  005512                  MOV     #DRVLST,R0      ;INIT DRIVE LIST POINTER
020270  005001                          CLR     R1              ;INIT DRIVE COUNTER
020272  005037  007426                  CLR     OVRAVG          ;INIT SUMS TO 0
020276  005037  007430                  CLR     COMAVG
020302  012702  006612                  MOV     #NOSCR0,R2      ;SET POINTER TO OVRWRT SCORES
020306  012703  007112                  MOV     #NCSCR0,R3      ;SET POINTER TO READ SCORES
020312  005720                  6$:     TST     (R0)+           ;COMPUTE TOTAL NUMBER OF DRIVES
020314  001402                          BEQ     8$              ;    IN R1
020316  005201                          INC     R1
020320  000774                          BR      6$
020322  010100                  8$:     MOV     R1,R0           ;PUT NO. OF DRIVES IN R0
020324  010204                          MOV     R2,R4           ;SET POINTERS TO SCORES FOR THIS TRACK
020326  010305                          MOV     R3,R5
020330  062437  007426          9$:     ADD     (R4)+,OVRAVG    ;ADD SCORES TO SUMS
020334  066437  000136  007426          ADD     94.(R4),OVRAVG
020342  062537  007430                  ADD     (R5)+,COMAVG
020346  066537  000136  007430          ADD     94.(R5),COMAVG
020354  005300                          DEC     R0              ;SEE IF ALL SCORES ADDED FOR THIS TRACK
020356  001364                          BNE     9$              ;BR IF NOT YET
020360  062702  000040                  ADD     #32.,R2         ;POINT TO SCORES FOR NEXT TRACK
020364  062703  000040                  ADD     #32.,R3
020370  020227  006712                  CMP     R2,#NOSCR0+64.  ;SEE IF ALL TRACKS DONE YET
020374  101752                          BLOS    8$              ;BR IF NOT YET
                                ;DIVIDE TO GET AVERAGES
                                        CLR     R4              ;GET DIVISOR IN R4-R5
                                        CLR     R5
                                        ADD     R1,R5
                                        ADD     R1,R5
                                        ADD     R1,R5
                                        ASL     R5
020412  012737  007426  001304          MOV     #OVRAVG,$TMP11  ;SET POINTER TO OVRWRT SCORE SUM
020420  005037  001306                  CLR     $TMP12          ;CLEAR SCORE INDICATOR
020424  005000                  10$:    CLR     R0              ;GET DIVIDEND IN R0-R1-R2-R3
020426  005001                          CLR     R1
020430  005002                          CLR     R2
020432  017703  160646                  MOV     @$TMP11,R3
020436  004737  042676                  JSR     PC,M.DPID       ;DIVIDE TO GET AVERAGE
020442  010500                          MOV     R5,R0           ;GET DIVISOR
020444  006200                          ASR     R0              ;DIVIDE IT BY 2
020446  020100                          CMP     R1,R0           ;SEE IF REMNDR > HALF DIVISOR
020450  103401                          BLO     12$             ;BR IF NOT
```

```
4673  020452  005203                          INC     R3                      ;ADD 1 TO QUOTIENT
4674  020454  010377  160624                  MOV     R3,@$TMP11              ;STORE AVERAGE SCORE
4675  020460  012737  007430  001304  12$:    MOV     #COMAVG,$TMP11          ;SET POINTER FOR READ SCORE SUM
4676  020466  005137  001306                  COM     $TMP12                  ;COMPLEMENT SCORE INDICATOR
4677  020472  001354                          BNE     10$                     ;BR TO COMPUTE READ SCORE AVERAGE
4678  020474  104410                          RESREG                          ;RESTORE R0-R5
4679  020476  000207                          RTS     PC                      ;RETURN
4680
4681
4682
4683                                  ;;*****************************************************************
4684                                  ;*NEDHDL - NON-EXISTENT DRIVE HANDLER
4685                                  ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
4686                                  ;*   WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
4687                                  ;*     (NED = NON-EXISTENT DRIVE)
4688                                  ;*****************************************************************
4689  020500  032765  010000  000020  NEDHDL: BIT     #NED,P.CS2(R5)          ;SEE IF NED ON DRIVE SELECT
4690  020506  001002                          BNE     1$                      ;BR IF NED SET
4691  020510  000137  031504                  JMP     ERRHDL                  ;GO HANDLE  OTHER ERROR
4692  020514  010046                  1$:     MOV     R0,-(SP)                ;SAVE R0,R1
4693  020516  010146                          MOV     R1,-(SP)
4694  020520  012700  000001                  MOV     #1,R0                   ;SET BIT POINTER
4695  020524  005001                          CLR     R1                      ;CLEAR COUNTER
4696  020526  120137  005430          2$:     CMPB    R1,DRIVE                ;SEE IF R1 = CURRENT DRIVE NUMBER
4697  020532  001403                          BEQ     3$                      ;BR IF  =
4698  020534  005201                          INC     R1                      ;INCREMENT COUNTER
4699  020536  006300                          ASL     R0                      ;SHIFT BIT POINTER
4700  020540  000772                          BR      2$                      ;TRY AGAIN
4701  020542  040037  005476          3$:     BIC     R0,NEWON                ;CLEAR ONLINE BIT FOR  THIS DRIVE
4702  020546  012601                          MOV     (SP)+,R1                ;RESTORE R0,R1
4703  020550  012600                          MOV     (SP)+,R0
4704  020552  000137  033754                  JMP     RETNML                  ;TAKE  NORMAL  RETURN
4705
4706
4707
4708
4709
4710                                  ;;*****************************************************************
4711                                  ;*DCKHDL - DATA CHECK ERROR HANDLER
4712                                  ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER, WHICH IS USED
4713                                  ;*WHEN DCK ERROR IS EXPECTED (AND VALID).
4714                                  ;*IF A HEADER VRC ERROR OCCURS, IT IS HANDLED, ALSO.
4715                                  ;*****************************************************************
4716  020556  005037  005424          DCKHDL: CLR     RECODE                  ;CLEAR ERROR FLAGS
4717  020562  032765  000400  000034          BIT     #HVRC,P.ER(R5)          ;SEE IF HEADER VRC ERROR
4718  020570  001404                          BEQ     4$                      ;BR IF NOT
4719  020572  052737  000004  005424          BIS     #HVRCER,RECODE          ;SET HVRC ERROR FLAG
4720  020600  000411                          BR      8$                      ;EXIT
4721  020602  032765  100000  000034  4$:     BIT     #DCK,P.ER(R5)           ;SEE IF DCK ERROR OCCURRED
4722  020610  001002                          BNE     6$                      ;BR IF YES
4723  020612  000137  031504                  JMP     ERRHDL                  ;GO HANDLE OTHER ERROR
4724  020616  052737  000020  005424  6$:     BIS     #DCKERR,RECODE          ;SET DCK ERROR FLAG
4725  020624  105237  003144          8$:     INCB    DCEFLG                  ;INCREMENT DCK ERROR FLAG
4726  020630  000137  033754                  JMP     RETNML                  ;TAKE NORMAL RETURN FROM DRIVER
4727
4728
```

```
                                ;;*****************************************************************
                                .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
                                ;;*****************************************************************
020634  010146                KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
020636  017701  160304              MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
020642  042701  177600              BIC     #177600,R1      ;MASK OUT UNUSED BITS
020646  120127  000172              CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
020652  003005                      BGT     20$             ;BR IF NOT
020654  120127  000141              CMPB    R1,#141
020660  002402                      BLT     20$             ;BR IF NOT
020662  042701  000040              BIC     #BIT5,R1        ;MAKE IT UPPER CASE
020666  010137  005442        20$:  MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
020672  122701  000003              CMPB    #003,R1         ;SEE IF (↑C) TYPED
020676  001007                      BNE     2$              ;BR IF NOT (↑C)
020700  104401  012102              TYPE    ,CNTRLC         ;ECHO (↑C)
020704  042777  000100  160232  1$: BIC     #BIT6,@$TKS     ;CLEAR KBD INTERRUPT ENABLE BIT
020712  012601                      MOV     (SP)+,R1        ;RESTORE R1
020714  000002                      RTI                     ;RETURN
020716  122701  000032        2$:   CMPB    #032,R1         ;SEE IF (↑Z) TYPED
020722  001003                      BNE     3$              ;BR IF NOT (↑Z)
020724  104401  012107              TYPE    ,CNTRLZ         ;ECHO (↑Z)
020730  000765                      BR      1$              ;RETURN
020732  122701  000022        3$:   CMPB    #022,R1         ;SEE IF (↑R) TYPED
020736  001003                      BNE     4$              ;BR IF NOT (↑R)
020740  104401  012114              TYPE    ,CNTRLR         ;ECHO (↑R)
020744  000757                      BR      1$              ;RETURN
020746  122701  000007        4$:   CMPB    #007,R1         ;SEE IF (↑G) TYPED
020752  001003                      BNE     5$              ;BR IF NOT (↑G)
020754  104401  012126              TYPE    ,CNTRLG         ;ECHO (↑G)
020760  000751                      BR      1$              ;RETURN
020762  104401  005442        5$:   TYPE    ,INTCHR         ;ECHO INPUT
020766  104401  001325              TYPE    ,$CRLF          ;DO <CR> AND <LF>
020772  000744                      BR      1$              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  PREPKB - PREPARE FOR KEYBOARD INPUT
                                ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
                                ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
                                ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
                                ;*ENABLES KBD INTERRUPT.
                                ;* CALL:
                                ;*      JSR     PC,PREPKB
                                ;;*****************************************************************

020774  005077  160146        PREPKB: CLR     @$TKB           ;CLEAR KBD BUFFER AND DONE BIT
021000  005037  005442                CLR     INTCHR          ;CLEAR TTY INPUT WORD
021004  052777  000100  160132        BIS     #BIT6,@$TKS     ;ENABLE KBD INTERRUPT
021012  000207                        RTS     PC              ;RETURN


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

```
                                .SBTTL  ECOBAD - ECHO BAD TTY INPUT
                                ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
                                ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
                                ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
                                ;*AND <LF> ARE DONE.
                                ;* CALL -     JSR      PC,ECOBAD
                                ;;*****************************************************************

021014  104401  005442         ECOBAD: TYPE    ,INTCHR         ;ECHO BAD CHARACTER
021020  104401  001324                 TYPE    ,$QUES          ;TYPE <?> AND <CR>, <LF>
021024  000207                         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.
                                ;;*****************************************************************
021026  016646  000002         GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
021032  104403                         TYPOS                   ;TYPE IT
021034     006                         .BYTE   6               ;SIX DIGITS
021035     000                         .BYTE   0               ;SUPPRESS LEADING ZEROS
021036  104401  011254                 TYPE    ,NEWMSG         ;TYPE    "    NEW = "
021042  004737  023424                 JSR     PC,RDCHRS       ;READ NEW VALUE FROM KBD
021046  021100                         7$                      ;(↑C) RETURN ADDRESS
021050  021100                         7$                      ;(↑Z) RETURN ADDRESS
021052  021100                         7$                      ;(↑U) OR ERROR RETURN ADDRESS
021054  005700                         TST     R0              ;SEE IF ANY CHARS TYPED
021056  001001                         BNE     4$              ;BR IF YES
021060  000207                         RTS     PC              ;RETURN - OLD VALUE UNCHANGED
021062  020027  000006         4$:     CMP     R0,#6           ;SEE IF > 6 CHARS TYPED
021066  003407                         BLE     8$              ;BR IF NOT BAD
021070  104401  005212         6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
021074  104401  001324                 TYPE    ,$QUES
021100  162716  000010         7$:     SUB     #10,(SP)        ;FIX ERROR RETURN PC
021104  000207                         RTS     PC              ;ERROR RETURN
021106  012746  005212         8$:     MOV     #BUFFO,-(SP)    ;PUT POINTER TO CHARS ON STACK
021112  004737  042116                 JSR     PC,OCTBIN       ;CONVERT DIGITS TO BINARY
021116  021070                         6$                      ;ERROR RETURN ADDRESS
021120  012600                         MOV     (SP)+,R0        ;GET NEW BINARY VALUE
021122  005737  042250                 TST     $HIOCT          ;SEE IF HI BITS ARE 0
021126  001360                         BNE     6$              ;BR IF NOT
021130  010066  000002                 MOV     R0,2(SP)        ;PUT NEW VALUE ON STACK
021134  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.
                                        ;;*****************************************************************
021136  022737  000176  001140  GTSWRG: CMP     #SWREG,SWR      ;SEE IF SOFTWARE SWR SELECTED
021144  001010                          BNE     6$              ;BR IF NOT
021146  013746  000176                  MOV     SWREG,-(SP)     ;PUT OLD VALUE ON STACK
021152  104401  011245                  TYPE    ,SWRMSG         ;TYPE "SWR = "
021156  004737  021026                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
021162  012637  000176                  MOV     (SP)+,SWREG     ;STORE NEW VALUE
021166  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
                                        ;;*****************************************************************
021170  012737  031504  003056  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
021176  012737  030272  003054          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
021204  013702  003046                  MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
021210  012705  002640                  MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
021214  105037  003150                  CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
021220  004737  021270                  JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
021224  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
021232  004737  030146                  JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
021236  113765  005430  000000          MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
021244  113737  005430  002724          MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
021252  113765  003134  000007          MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
021260  153765  003133  000007          BISB    TYPFMT,P.CS1H(R5)       ;SET DRV TYPE. 0=RK06, 4=RK07
021266  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
                                        ;;*****************************************************************
021270  010046                  CLRPRM: MOV     R0,-(SP)        ;SAVE R0
021272  010546                          MOV     R5,-(SP)        ;SAVE R5
021274  010500                          MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
021276  062705  000016                  ADD     #P.CS1,R5       ;COMPUTE LIMIT ADDRESS
021302  005020                  1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
```

```
4897  021304  020005                  CMP     R0,R5           ;SEE IF DONE YET
4898  021306  001375                  BNE     1$              ;BR IF  NOT DONE  YET
4899  021310  012605                  MOV     (SP)+,R5        ;RESTORE R5
4900  021312  012600                  MOV     (SP)+,R0        ;RESTORE R0
4901  021314  000207                  RTS     PC              ;RETURN
4902
4903
4904
4905                          ;;*****************************************************************
4906                          ;.SBTTL  SCNDRV - SCAN DRIVE FOR STATUS
4907                          ;*THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
4908                          ;*THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
4909                          ;*AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
4910                          ;*OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
4911                          ;*MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
4912                          ;*A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
4913                          ;*AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
4914                          ;*MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
4915                          ;* CALL -    JSR     PC,SCNDRV
4916                          ;*              <ERROR RETURN ADDRESS>
4917                          ;*
4918                          ;;*****************************************************************
4919
4920  021316  004737  021170  SCNDRV: JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
4921  021322  042712  000100  3$:     BIC     #IE,(R2)        ;DISABLE RK06 INTERRUPT
4922  021326  113762  005430  000010  MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
4923  021334  105737  003133          TSTB    TYPFM†          ;SEE IF RK07
4924  021340  001003                  BNE     5$              ;BR IF YES
4925  021342  012712  000001          MOV     #GO,(R2)        ;ELSE SET GO BIT FOR RK06 IN RKCS1
4926  021346  000402                  BR      7$
4927
4928  021350  012712  002001  5$:     MOV     #<CDT!GO>,(R2)  ;SET GO FOR RK07
4929  021354  005037  005444  7$:     CLR     SCRACH          ;CLEAR STALL COUNTER
4930  021360  005237  005444  15$:    INC     SCRACH          ;INCREMENT STALL COUNTER
4931  021364  001375                  BNE     15$             ;STALL FOR SEVERAL MILLI-SEC
4932  021366  032762  001000  000010  BIT     #MDS,RKCS2(R2)  ;SEE IF MDS ERROR
4933  021374  001417                  BEQ     18$             ;BR IF NOT
4934  021376  112765  000101  000001  MOVB    #SELDRV,P.CMND(R5) ;SET CMND FOR ERROR REPORT
4935  021404  011265  000016          MOV     (R2),P.CS1(R5)  ;GET RKCS1 FOR REPORT
4936  021410  004737  037644          JSR     PC,I.CST1       ;GET OTHER RK611 REGS FOR REPORT
4937  021414  004737  031224          JSR     PC,REPSUP       ;SET UP ERROR REPORT
4938  021420  104052                  ERROR   52              ;REPORT MDS ERROR
4939  021422  052737  000200  005424  BIS     #ABORT,RECODE   ;SET ABORT FLAG
4940  021430  000137  033336          JMP     ALLTRM          ;GO ABORT TESTING
4941  021434  004737  021170  18$:     JSR     PC,INITSS       ;INIT THE S.S.
4942  021440  112765  000141  000001  MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
4943  021446  012737  020500  003056  MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
4944  021454  012737  000377  005476  MOV     #377,NEWON      ;INIT. ON-LINE INDICATOR
4945  021462  004737  030146          JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
4946  021466  112737  000040  011312  MOVB    #40,BADDRV+7    ;PUT A SPACE IN MSG BUF
4947                          ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
4948  021474  012737  031504  003056  MOV     #ERRHDL,A.ABNL  ;RESTORE ABNL RETURN ADDRESS
4949  021502  022737  000377  005476  CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
4950  021510  001425                  BEQ     4$              ;BR IF NED NOT SET
4951  021512  113737  005430  011311  MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
4952  021520  152737  000060  011311  BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
```

```
 4953  021526  104401  011303                    TYPE    ,BADDRV          ;TYPE "DRIVE X"
 4954  021532  104401  011314                    TYPE    ,NXDRIV          ;TYPE "NON-EXISTENT"
 4955  021536  132737  000200  001351            BITB    #BIT7,$ENVM      ;SEE IF UNDER APT
 4956  021544  001401                            BEQ     2$               ;BR IF NO
 4957  021546  104123                            ERROR   123              ;NED UNDER APT SIZING
 4958                        ;SERVICE ERRORS HERE
 4959  021550  042762  000100  000000  2$:       BIC     #IE,RKCS1(R2)    ;DISABLE RK06 INTERRUPT
 4960  021556  017616  000000                    MOV     @(SP),(SP)       ;SET UP ERROR RETURN ADDRESS
 4961  021562  000207                            RTS     PC               ;ERROR RETURN
 4962                        ;SEE IF DRIVE IS READY
 4963  021564  032765  000200  000040  4$:       BIT     #S.DRY,P.AOO(R5) ;TEST FOR DRIVE READY
 4964  021572  001013                            BNE     6$               ;BR IF DRIVE IS READY
 4965  021574  113737  005430  011311            MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO. INTO BUF
 4966  021602  152737  000060  011311            BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
 4967  021610  104401  011303                    TYPE    ,BADDRV          ;TYPE "DRIVE X"
 4968  021614  104401  011333                    TYPE    ,NTREDY          ;TYPE "NOT READY"
 4969  021620  000753                            BR      2$               ;TAKE ERROR EXIT
 4970                        ;SEE IF DRIVE IS WRITE ENABLED
 4971  021622  032765  004000  000040  6$:       BIT     #S.WRL,P.AOO(R5) ;SEE IF WRITE LOCK SET
 4972  021630  001413                            BEQ     8$               ;BR IF WRITE LOCK NOT SET
 4973  021632  113737  005430  011311            MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO.
 4974  021640  152737  000060  011311            BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
 4975  021646  104401  011303                    TYPE    ,BADDRV          ;TYPE "DRIVE X"
 4976  021652  104401  011347                    TYPE    ,WRTLOK          ;TYPE "WRITE-LOCKED"
 4977  021656  000734                            BR      2$               ;TAKE ERROR EXIT
 4978                        ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
 4979  021660  112765  000103  000001  8$:       MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
 4980  021666  004737  030146                    JSR     PC,DRVCAL        ;SET VOLUME VALID
 4981  021672  112765  000113  000001            MOVB    #RECAL,P.CMND(R5)        ;SET RECAL CMND
 4982  021700  004737  030146                    JSR     PC,DRVCAL        ;RECAL THE DRIVE
 4983  021704  112765  000121  000001            MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 4984  021712  013765  024404  000002            MOV     LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)/1456(8)
 4985  021720  112765  000002  000005            MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
 4986  021726  012765  053342  000010            MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
 4987  021734  012765  177774  000012            MOV     #-4,P.WC(R5)     ;READ 4 WORDS
 4988  021742  142765  000020  000007            BICB    #B.CFMT,P.CS1H(R5)       ;SET 22 SECTOR FORMAT
 4989  021750  153765  003133  000007            BISB    TYPFMT,P.CS1H(R5)
 4990  021756  004737  030146                    JSR     PC,DRVCAL        ;READ 4 WORDS OF BAD SECTOR FILE
 4991  021762  032737  100000  005424            BIT     #ANYDER,RECODE   ;SEE IF DATA ERROR
 4992  021770  001402                            BEQ     10$              ;BR IF OK
 4993  021772  104401  012036                    TYPE    ,BAD632          ;TYPE READ ERROR MESSAGE
 4994  021776  022737  177777  053350  10$:      CMP     #177777,RWBUF+6  ;SEE IF ALL 1'S IN I.D. WORD 3
 4995  022004  001013                            BNE     12$              ;BR IF NOT ALL 1'S
 4996  022006  113737  005430  011311            MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO.
 4997  022014  152737  000060  011311            BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
 4998  022022  104401  011303                    TYPE    ,BADDRV          ;TYPE "DRIVE X"
 4999  022026  104401  011366                    TYPE    ,ALNPAK          ;TYPE "LOADED WITH ALIGN PACK"
 5000  022032  000646                            BR      2$               ;TAKE ERROR EXIT
 5001                        ;ERROR FREE RETURN
 5002  022034  062716  000002            12$:    ADD     #2,(SP)          ;FIX UP RETURN PC
 5003  022040  000207                            RTS     PC               ;RETURN
 5004
 5005
 5006                        ;;*****************************************************************
 5007                        ;*SETTUP - SET UP FOR LOOP ON ERROR
 5008                        ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
```

```
                                        ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
                                        ;*************************************************************
022042  011637  001076                  SETTUP: MOV     (SP),@#STACK-2  ;MOVE RETURN PC ON STACK
022046  012706  001076                          MOV     #STACK-2,SP     ;RE-INIT THE STACK POINTER
022052  005037  005424                          CLR     RECODE          ;CLEAR ERROR RECOVERY FLAGS
022056  105037  003135                          CLRB    ERRCNT          ;CLEAR RETRY ERROR COUNT
022062  012705  002640                          MOV     #PARMO,R5       ;SET PARAM BLK ADRS
022066  112765  000177  000001                  MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
022074  004737  030146                          JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
022100  012737  000000  177776                  MOV     #PRO,@#PS       ;RE-ESTABLISH PRIORITY 0
022106  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.
                                        ;*************************************************************
022110                                  MTCART:
022110  013700  005554                          MOV     DRVPTR,R0       ;GET POINTER TO FIRST DRIVE FOR THIS SUBSYS
022114  112037  010366                          MOVB    (R0)+,MNTPAK+22. ;PUT DRIVE NAME INTO MSG
022120  111037  010365                          MOVB    (R0),MNTPAK+21.
022124  162700  005512                          SUB     #DRVLST,R0      ;SUBTRACT LIST ADRS FROM POINTER
022130  006200                                  ASR     R0              ;DIVIDE BY TWO TO GET DRIVE NO.
022132  110037  003154                          MOVB    R0,LOGDRV       ;STORE LOGICAL DRIVE NUMBER
022136  004737  021170                          JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
022142  042762  000100  000000                  BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT FOR MAN. INTERVENTION
022150  104401  010340                  10$:    TYPE    ,MNTPAK         ;TYPE MOUNT PACK MSG
022154  104401  010602                  5$:     TYPE    ,TYPRWN         ;TYPE "TYPE R<CR> WHEN "
022160  104401  010623                          TYPE    ,DRIRDY         ;TYPE "DRIVE READY :"
022164  004737  023424                          JSR     PC,RDCHRS       ;WAIT FOR R<CR> RESPONSE
022170  022214                                  12$                     ;(↑C) RETURN ADDRESS
022172  022236                                  16$                     ;(↑Z) RETURN ADDRESS
022174  022154                                  5$                      ;(↑U) RETURN ADDRESS
022176  005700                                  TST     R0              ;SEE IF NULL INPUT
022200  001025                                  BNE     8$              ;BR IF NOT NULL
022202  104401  005212                  6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
022206  104401  001324                          TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
022212  000760                                  BR      5$              ;GO ASK AGAIN
022214  012706  001100                  12$:    MOV     #STACK,SP       ;RE-INIT THE STACK
022220  105737  003126                          TSTB    MDFLAG          ;SEE IF 200 START
022224  001002                                  BNE     14$             ;BR IF NOT
022226  000137  012176                          JMP     DFSTRT          ;RESTART PROGRAM - 200 START
022232  000137  013002                  14$:    JMP     ALLSYS          ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
022236  105737  003126                  16$:    TSTB    MDFLAG          ;SEE IF 200 START
022242  001744                                  BEQ     5$              ;BR IF YES, TO ASK AGAIN
022244  012706  001100                          MOV     #STACK,SP       ;RE-INIT THE STACK
022250  000137  013446                          JMP     ASKSYS          ;GO ASK FOR SUBSYS TO TEST AGAIN
022254  023727  005212  000122  8$:     CMP     BUFFO,#'R       ;SEE IF R<CR> TYPED
022262  001347                                  BNE     6$              ;BR IF NOT
```

```
5065   022264   004737   021170                    JSR     PC,INITSS       ;INITIALIZE S.S.
5066                                      ;CHECK STATUS OF DRIVE TO BE TESTED NEXT
5067   022270   117737   163260   005430          MOVB    @DRVPTR,DRIVE   ;SET DRIVE NUMBER
5068   022276   142737   000060   005430          BICB    #'0,DRIVE       ;STRIP THE ASCII
5069   022304   004737   021316                    JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE -
5070   022310   022150                            10$                     ;   IF NOT VALID, TYPE MSG AND RETURN
5071                                                                      ;   TO WAIT FOR R<CR> AGAIN
5072                                      ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
5073   022312   004737   021170                    JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
5074   022316   112765   000125   000001          MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
5075   022324   004737   030146                    JSR     PC,DRVCAL       ;DO A READ HEADER
5076   022330   013762   024406   000000          MOV     HOLD1,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 0
5077   022336   032762   000200   000000  24$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
5078   022344   001774                            BEQ     24$
5079   022346   013762   024410   000000          MOV     HOLD2,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
5080   022354   032762   000200   000000  26$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
5081   022362   001774                            BEQ     26$
5082   022364   105037   003134                    CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
5083   022370   005762   000024                    TST     RKDB(R2)        ;POP SILO ONE TIME
5084   022374   032762   001000   000024          BIT     #BIT9,RKDB(R2)  ;TEST FORMAT BIT IN HEADER WORD 2
5085   022402   001403                            BEQ     28$             ;BR IF 22 SECTOR FORMAT
5086   022404   152737   000020   003134          BISB    #8.CFMT,FORMAT  ;SET 20 SECTOR FORMAT FOR THIS DRIVE
5087   022412   004737   021170           28$:    JSR     PC,INITSS       ;INIT THE S.S.
5088   022416   004737   026540                    JSR     PC,REDBSF       ;READ BAD SECTOR FILE FOR THIS DRIVE
5089   022422   105737   003126                    TSTB    MDFLAG          ;SEE IF 200 START
5090   022426   001004                            BNE     32$             ;BR IF NOT 200 START
5091   022430   122737   000061   003155          CMPB    #'1,PASSNO      ;SEE IF FIRST PASS
5092   022436   001012                            BNE     30$             ;BR IF NOT FIRST PASS
5093   022440   004737   023736           32$:    JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
5094   022444   004737   024056                    JSR     PC,CRTSER       ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
5095   022450   032777   000200   156462          BIT     #BIT7,@SWR      ;SEE IF BAD SECTORS SHOULD BE TYPED
5096   022456   001402                            BEQ     30$             ;BR IF NOT
5097   022460   004737   027030                    JSR     PC,TYPBSF       ;TYPE BAD SECTOR FILES
5098   022464   005037   005442           30$:    CLR     INTCHR          ;INIT. TTY INPUT CHAR BUFFER
5099   022470   000207                            RTS     PC              ;RETURN
5100
5101
5102
5103                                      ;;*****************************************************************
5104                                      .SBTTL  DMCART - DISMOUNT TEST CARTRIDGE
5105                                      ;*THIS SUBROUTINE DIRECTS THE OPERATOR IN THE DISMOUNTING
5106                                      ;*OF THE TEST CARTRIDGE ON THE DRIVE CURRENTLY UNDER TEST.
5107                                      ;;*****************************************************************
5108   022472                             DMCART:
5109   022472   013700   005554                    MOV     DRVPTR,R0       ;GET DRIVE LIST POINTER
5110   022476   112037   010423                    MOVB    (R0)+,DISPAK+15. ;PUT DRIVE NAME INTO MSG
5111   022502   111037   010422                    MOVB    (R0),DISPAK+14.
5112   022506   004737   021170                    JSR     PC,INITSS       ;INITIALIZE SUBSYS
5113   022512   042762   000100   000000          BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTRPT FOR MANUAL INTERVENTION
5114   022520   104401   010404                    TYPE    ,DISPAK         ;TYPE "UNLOAD DRIVE XX AND REMOVE PACK"
5115   022524   104401   010602           5$:     TYPE    ,TYPRWN         ;TYPE "TYPE R<CR> WHEN "
5116   022530   104401   010642                    TYPE    ,DRIDON         ;TYPE "DONE :"
5117   022534   004737   023424                    JSR     PC,RDCHRS       ;WAIT FOR R<CR> RESPONSE
5118   022540   022564                            12$                     ;(↑C) RETURN ADDRESS
5119   022542   022606                            16$                     ;(↑Z) RETURN ADDRESS
5120   022544   022524                            5$                      ;(↑U) RETURN ADDRESS
```

```
5121   022546  005700                          TST     R0              ;SEE IF NULL INPUT
5122   022550  001025                          BNE     8$              ;BR IF NOT NULL
5123   022552  104401  005212          6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
5124   022556  104401  001324                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5125   022562  000760                          BR      5$              ;GO ASK AGAIN
5126   022564  012706  001100          12$:    MOV     #STACK,SP       ;RE-INIT THE STACK
5127   022570  105737  003126                  TSTB    MDFLAG          ;SEE IF 200 START
5128   022574  001002                          BNE     14$             ;BR IF NOT
5129   022576  000137  012176                  JMP     DFSTRT          ;RESTART PROGRAM - 200 START
5130   022602  000137  013002          14$:    JMP     ALLSYS          ;GO ASK FOR SUBSYS'S AND DRIVES AGAIN
5131   022606  105737  003126          16$:    TSTB    MDFLAG          ;SEE IF 200 START
5132   022612  001744                          BEQ     5$              ;BR IF YES, TO ASK AGAIN
5133   022614  012706  001100                  MOV     #STACK,SP       ;RE-INIT THE STACK
5134   022620  000137  013446                  JMP     ASKSYS          ;GO ASK FOR SUBSYS TO TEST AGAIN
5135   022624  023727  005212  000122  8$:     CMP     BUFFO,#'R       ;SEE IF R<CR> TYPED
5136   022632  001347                          BNE     6$              ;BR IF NOT
5137   022634  004737  021170                  JSR     PC,INITSS       ;INITIALIZE S.S.
5138   022640  000207                          RTS     PC              ;RETURN
5139
5140
5141
5142                                    ;;*****************************************************************
5143                                    .SBTTL  INTBLD - INITIALIZE TABLED
5144                                    ;*THIS SUBROUTINE LOADS TABLED WITH THE INITIAL PATTERN OF DRIVE NUMBERS
5145                                    ;*WHICH IS THE CYLINDER BLOCK LAYOUT FOR THE FIRST CURRENT ZONE. IT USES
5146                                    ;*THE STARTING DRIVE NUMBER LIST, STDLST, FOR THE FIRST COLUMN, AND
5147                                    ;*INCREMENTS THESE DRIVE NUMBERS ACROSS THE TABLE, FORMING ROWS. FOR
5148                                    ;*ANY DRIVE WHICH IS NOT PRESENT, THE ENTRY IS SET = 377.
5149                                    ;;*****************************************************************
5150   022642  104407                  INTBLD: SAVREG                  ;SAVE R0-R5
5151                                    ;LOAD THE FIRST COLUMN OF TABLED FROM STDLST
5152   022644  012700  006552                  MOV     #STDLST,R0      ;STARTING ADRS OF STDLST
5153   022650  012701  006112                  MOV     #TABLED,R1      ;STARTING ADRS OF TABLED
5154   022654  012703  000020                  MOV     #16.,R3         ;SET COUNTER =  16
5155   022660  112011          4$:     MOVB    (R0)+,(R1)      ;LOAD A BYTE INTO TABLED
5156   022662  062701  000020                  ADD     #16.,R1         ;INCREMENT TABLED POINTER
5157   022666  005303                          DEC     R3              ;DECR COUNTER
5158   022670  001373                          BNE     4$              ;BR IF NOT DONE YET
5159                                    ;GET INDEX OF FIRST ZERO ENTRY IN DRVLST INTO R1
5160   022672  012700  005512                  MOV     #DRVLST,R0      ;STARTING ADRS OF DRVLST
5161   022676  005001                          CLR     R1              ;INIT INDEX TO 0
5162   022700  005720          6$:     TST     (R0)+           ;SEE IF ENTRY =  0
5163   022702  001404                          BEQ     8$              ;BR IF YES
5164   022704  005201                          INC     R1              ;INCREMENT POINTER
5165   022706  020127  000020                  CMP     R1,#16.         ;SEE IF DONE LOOKING YET
5166   022712  002772                          BLT     6$              ;BR IF NOT DONE YET
5167                                    ;INCREMENT THE DRIVE NOS. TO FORM THE ROWS
5168   022714  012700  006112          8$:     MOV     #TABLED,R0      ;INIT TABLED POINTER
5169   022720  012704  000017          16$:    MOV     #15.,R4         ;INIT ROW ELEMENT COUNTER
5170   022724  112003                  18$:    MOVB    (R0)+,R3        ;INIT ROW ELEMENT VALUE
5171   022726  005203                          INC     R3              ;INCR ROW ELEMENT VALUE
5172   022730  020327  000020                  CMP     R3,#20          ;SEE IF ROW ELEMENT IS VALID
5173   022734  002401                          BLT     20$             ;BR IF VALUE IS OK
5174   022736  005003                          CLR     R3              ;RESET VALUE TO 0
5175   022740  020301          20$:    CMP     R3,R1           ;SEE IF VALUE IS > ACTUAL DRIVES
5176   022742  002403                          BLT     22$             ;BR IF VALUE OK
```

```
5177   022744   112720   000377                MOVB    #377,(R0)+      ;SET INVALID ENTRY = 377
5178   022750   000401                         BR      24$             ;PROCEED
5179   022752   110320                 22$:    MOVB    R3,(R0)+        ;SET VALUE IN TABLE
5180   022754   005304                 24$:    DEC     R4              ;DECR ROW ELEMENT COUNTER
5181   022756   001363                         BNE     18$             ;BR IF NOT DONE WITH ROW YET
5182   022760   020027   006512                CMP     R0,#TABLED+256. ;SEE IF DONE WITH TABLE YET
5183   022764   103755                         BLO     16$             ;BR IF NOT DONE YET
5184                                  ;SET INVALID ELEMENTS OF FIRST COLUMN = 377
5185   022766   012700   006112                MOV     #TABLED,R0      ;INIT COLUMN POINTER
5186   022772   012703   000020                MOV     #16.,R3         ;INIT COL. ELEMENT COUNTER
5187   022776   121001                 26$:    CMPB    (R0),R1         ;SEE IF ELEMENT IS VALID
5188   023000   002402                         BLT     30$             ;BR IF YES
5189   023002   112710   000377                MOVB    #377,(R0)       ;SET INVALID ELEMENT = 377
5190   023006   062700   000020         30$:    ADD     #16.,R0         ;INCREMENT COLUMN POINTER
5191   023012   005303                         DEC     R3              ;SEE IF ALL DONE YET
5192   023014   001370                         BNE     26$             ;BR IF NOT ALL DONE YET
5193   023016   104410                         RESREG                  ;RESTORE R0-R5
5194   023020   000207                         RTS     PC              ;RETURN
5195
5196
5197
5198                                  ;;*****************************************************************
5199                                  .SBTTL  ROTBLD - ROTATE TABLED RIGHT 3 COLUMNS
5200                                  ;*THIS SUBROUTINE ROTATES ALL ELEMENTS OF TABLED RIGHT 3 COLUMNS,
5201                                  ;*SO THAT THE DRIVE DATA WILL BE ROTATED SEVERAL SECTORS IN EACH
5202                                  ;*CURRENT ZONE.
5203                                  ;;*****************************************************************
5204   023022   104407                 ROTBLD: SAVREG                  ;SAVE R0-R5
5205   023024   012700   006512                MOV     #TABLED+256.,R0 ;POINT TO END OF TABLED
5206   023030   010001                         MOV     R0,R1           ;GET A COPY IN R1
5207   023032   012704   000015         4$:     MOV     #13.,R4         ;INIT ROW ELEMENT COUNTER
5208   023036   114037   001304                MOVB    -(R0),$TMP11    ;SAVE LAST 3 ELEMENTS IN THIS ROW
5209   023042   114037   001306                MOVB    -(R0),$TMP12
5210   023046   114037   001310                MOVB    -(R0),$TMP13
5211   023052   114041                 6$:     MOVB    -(R0),-(R1)     ;MOVE AN ELEMENT RIGHT 3 PLACES
5212   023054   005304                         DEC     R4              ;DECR ROW ELEMENT COUNTER
5213   023056   001375                         BNE     6$              ;BR IF NOT DONE YET
5214   023060   113741   001304                MOVB    $TMP11,-(R1)    ;MOVE IN 3 SAVED ELEMENTS
5215   023064   113741   001306                MOVB    $TMP12,-(R1)
5216   023070   113741   001310                MOVB    $TMP13,-(R1)
5217   023074   020027   006112                CMP     R0,#TABLED      ;SEE IF DONE WITH TABLE YET
5218   023100   101354                         BHI     4$              ;BR IF NOT DONE YET
5219   023102   104410                         RESREG                  ;RESTORE R0-R5
5220   023104   000207                         RTS     PC              ;RETURN
5221
5222
5223
5224                                  ;;*****************************************************************
5225                                  .SBTTL  WRTBLK - WRITE ONE CYLINDER BLOCK
5226                                  ;*THIS SUBROUTINE WRITES ALL THE SECTORS FOR THE CURRENT DRIVE
5227                                  ;*INTO THE CYLINDER BLOCK WHOSE STARTING CYL NUMBER IS IN
5228                                  ;*P.CYLN(R5) ON ENTRY. THE DATA IS WRITTEN ON ALL  SURFACES
5229                                  ;*THE LOGICAL DRIVE NUMBER MUST BE IN LOGDRV ON ENTRY.
5230                                  ;*
5231                                  ;*IT CALLS WRTSEC TO ACTUALLY WRITE EACH 256-WORD SECTOR.
5232                                  ;;*****************************************************************
```

```
5233  023106  104407                  WRTBLK: SAVREG                          ;SAVE R0-R5
5234  023110  113700  003154                  MOVB    LOGDRV,R0               ;GET LOGICAL DRIVE NO.
5235  023114  006300                        ASL     R0                      ;CONVERT IT TO DATA PATTERN INDEX
5236  023116  016003  006512                MOV     TABLEG(R0),R3           ;GET DATA PATTERN WORD INTO R3
5237  023122  012701  006112                MOV     #TABLED,R1              ;GET TABLED POINTER
5238  023126  012704  000020                MOV     #16.,R4                 ;INIT CYLINDER COUNTER
5239  023132  010100                  4$:   MOV     R1,R0
5240  023134  123720  003154            6$:   CMPB    LOGDRV,(R0)+            ;SEE IF DRIVE LISTED HERE
5241  023140  001375                        BNE     6$                      ;BR IF NOT FOUND YET IN THIS ROW
5242  023142  005300                        DEC     R0                      ;BACK UP  BY 1 BYTE
5243  023144  160100                        SUB     R1,R0                   ;COMPUTE ROW POSITION
5244  023146  116065  006572  000004        MOVB    SECLST(R0),P.SECT(R5) ;GET SECTOR NO. FROM ROW POSITION
5245  023154  105065  000005                CLRB    P.TRCK(R5)              ;SET TRACK = 0
5246  023160  004737  025770            8$:   JSR     PC,WRTSEC               ;WRITE 256 WORDS AT THIS SECTOR
5247  023164  105265  000005                INCB    P.TRCK(R5)              ;INCR TRACK NO.
5248  023170  126527  000005  000002        CMPB    P.TRCK(R5),#2           ;SEE IF ALL TRACKS DONE YET
5249  023176  003770                        BLE     8$                      ;BR IF NOT DONE YET
5250  023200  062701  000020                ADD     #16.,R1                 ;POINT TO START OF NEXT ROW
5251  023204  005265  000002                INC     P.CYLN(R5)              ;INCR CYLINDER NO.
5252  023210  005304                        DEC     R4                      ;DECREMENT CYLINDER COUNTER
5253  023212  001347                        BNE     4$                      ;BR IF 16 CYLS NOT DONE YET
5254  023214  104410                        RESREG                          ;RESTORE R0-R5
5255  023216  000207                        RTS     PC                      ;RETURN
5256
5257
5258
5259                                  ;;*****************************************************************
5260                                  ;*CKSCOR - DECREMENT DRIVE SCORES ROUTINE
5261                                  ;*THIS SUBROUTINE CHECKS TO SEE IF THE PROGRAM IS CURRENTLY
5262                                  ;*READING OVRWRT OR COMPAT DATA WITH OFFSETS AND HANDLING DATA CHECK
5263                                  ;*ERRORS. IF SO, THE SCORE FOR THE DRIVE WHICH WROTE THE
5264                                  ;*FAILING DATA IS DECREMENTED.
5265                                  ;*ON ENTRY, THE BYTE LABELED "TSTTYP" HAS THE FOLLOWING
5266                                  ;*POSSIBLE VALUES :
5267                                  ;*       TSTTYP = 0  - NOT PRESENTLY READING WITH OFFSET
5268                                  ;*              = 1  - PRESENTLY READING OVRWRT DATA WITH OFFSET
5269                                  ;*              = 2  - PRESENTLY READING COMPAT DATA WITH OFFSET
5270                                  ;*              = 3  - PRESENTLY READING SELF TEST DATA WITH OFFSET
5271                                  ;;*****************************************************************
5272  023220  104407                  CKSCOR: SAVREG                          ;SAVE R0-R5
5273  023222  105737  003143                  TSTB    TSTTYP                  ;SEE IF CURRENTLY TESTING WITH OFFSET
5274  023226  001474                        BEQ     20$                     ;BR TO EXIT, IF NOT
5275  023230  032737  000024  005424        BIT     #DCKERR!HVRCER,RECODE ;SEE IF A DCK OR HVRC ERR OCCURRED
5276  023236  001470                        BEQ     20$                     ;BR IF NOT
5277                                  ;GET THE NO. OF THE DRIVE WHICH WROTE DATA WHICH CAUSED DCK OR HVRC ERR
5278  023240  004737  034310                JSR     PC,GTPKAD               ;COMPUTE PACK ADDRESS OF ERROR
5279  023244  122737  000003  003143        CMPB    #3,TSTTYP               ;SEE IF DOING SELF-TEST
5280  023252  001002                        BNE     4$                      ;BR IF NOT
5281  023254  005004                        CLR     R4                      ;SET DRIVE NO. = 0
5282  023256  000431                        BR      10$                     ;CONTINUE
5283  023260  013701  001174            4$:   MOV     $REG5,R1                ;GET CYL NO. OF ERROR
5284  023264  042701  177760                BIC     #177760,R1              ;CLEAR HI CYL BITS
5285  023270  006301                        ASL     R1                      ;CONVERT TO TABLED INDEX FOR ROW
5286  023272  006301                        ASL     R1
5287  023274  006301                        ASL     R1
5288  023276  006301                        ASL     R1
```

```
5289   023300   062701   006112                ADD     #TABLED,R1      ;COMPUTE POINTER TO TABLED ROW
5290   023304   005000                         CLR     R0              ;INIT SECLST INDEX TO 0
5291   023306   123760   001200   006572  6$:  CMPB    $REG7,SECLST(R0) ;SEE IF SECTOR IS LISTED IN SECLST
5292   023314   001405                         BEQ     8$              ;BR IF YES
5293   023316   005200                         INC     R0              ;INCR SECLST INDEX
5294   023320   020027   000020                CMP     R0,#16.         ;SEE IF WHOLE LIST CHECKED YET
5295   023324   002770                         BLT     6$              ;BR IF NOT YET
5296   023326   000434                         BR      20$             ;SECTOR NOT LISTED - EXIT, BECAUSE
5297                                                                   ;  FAILING DRIVE IS NOT BEING TESTED
5298   023330   060001                 8$:     ADD     R0,R1           ;COMPUTE TABLED POINTER TO FAILING DRIVE
5299   023332   111104                         MOVB    (R1),R4         ;GET ACTUAL DRIVE NO.
5300   023334   122704   000377                CMPB    #377,R4         ;SEE IF DRIVE IS NOT BEING TESTED
5301   023340   001427                         BEQ     20$             ;BR IF DRIVE NOT TESTED
5302   023342   012700   007432         10$:    MOV     #NSCOR0,R0      ;GET ADRS OF NEG SCRATCH SCORES
5303   023346   105737   003156                TSTB    OFSDIR          ;SEE WHICH OFST DIRECTION
5304   023352   001402                         BEQ     12$             ;BR IF NEG
5305   023354   012700   007512                MOV     #PSCOR0,R0      ;GET ADRS OF POS SCRATCH SCORES
5306   023360   105737   001176         12$:    TSTB    $REG6           ;SEE IF TRACK = 0
5307   023364   001410                         BEQ     14$             ;BR IF YES
5308   023366   062700   000020                ADD     #16.,R0         ;POINT TO SCORES FOR TRACK 1
5309   023372   123727   001176   000001       CMPB    $REG6,#1        ;SEE IF TRACK = 1
5310   023400   001402                         BEQ     14$             ;BR IF YES
5311   023402   062700   000020                ADD     #16.,R0         ;POINT TO SCORES FOR TRACK 2
5312   023406   060400                 14$:    ADD     R4,R0           ;COMPUTE POINTER TO SCORE FOR DRIVE
5313   023410   121037   003157                CMPB    (R0),OFSCNT     ;SEE IF THIS SCORE IS PERFECT SO FAR
5314   023414   001001                         BNE     20$             ;EXIT IF NOT
5315   023416   105310                         DECB    (R0)            ;DECREMENT SCORE FOR THIS DRIVE
5316   023420   104410                 20$:    RESREG                  ;RESTORE R0-R5
5317   023422   000207                         RTS     PC              ;RETURN
5318
5319
5320
5321                                    ;;*****************************************************************
5322                                    .SBTTL  RDCHRS - READ A STRING OF KBD INPUT CHARS
5323                                    ;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
5324                                    ;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
5325                                    ;*THEM IN BUFF0, TERMINATED BY A NULL (0) BYTE.
5326                                    ;*RUB-OUT AND (↑U) FEATURES ARE PROVIDED.
5327                                    ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
5328                                    ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
5329                                    ;*TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
5330                                    ;*TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
5331                                    ;*IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
5332                                    ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
5333                                    ;*IS TAKEN.
5334                                    ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
5335                                    ;*IN R0.
5336                                    ;*
5337                                    ;*  CALL -      JSR     PC,RDCHRS
5338                                    ;*              <CONTROL-C RETURN ADDRESS>
5339                                    ;*              <CONTROL-Z RETURN ADDRESS>
5340                                    ;*              <CONTROL-U OR ERROR RETURN ADDRESS>
5341                                    ;*              RETURN
5342                                    ;;*****************************************************************
5343
5344   023424                           RDCHRS:
```

```
5345  023424  010146                          MOV     R1,-(SP)        ;SAVE R1
5346  023426  010246                          MOV     R2,-(SP)        ;SAVE R2
5347  023430  005000                          CLR     R0              ;INITIALIZE CHARACTER COUNT
5348  023432  005001                          CLR     R1              ;INITIALIZE RUB-OUT INDICATOR
5349                                  ;READ A CHARACTER
5350  023434  104406                  2$:     RDCHR                   ;READ A CHARACTER
5351  023436  112602                          MOVB    (SP)+,R2        ;GET CHARACTER INTO R2
5352                                  ;CHECK FOR (↑C)
5353  023440  122702  000003                  CMPB    #003,R2         ;SEE IF (↑C) TYPED
5354  023444  001006                          BNE     4$              ;BR IF NOT (↑C)
5355  023446  104401  012102                  TYPE    ,CNTRLC         ;ECHO (↑C)
5356  023452  017666  000004  000004  3$:     MOV     @4(SP),4(SP)    ;PUT RETURN ADDRESS ON STACK
5357  023460  000523                          BR      24$             ;BR TO TAKE EXIT
5358                                  ;CHECK FOR (↑Z)
5359  023462  122702  000032          4$:     CMPB    #032,R2         ;SEE IF (↑Z) TYPED
5360  023466  001006                          BNE     6$              ;BR IF NOT (↑Z)
5361  023470  104401  012107                  TYPE    ,CNTRLZ         ;ECHO (↑Z)
5362  023474  062766  000002  000004          ADD     #2,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADR.
5363  023502  000763                          BR      3$              ;BR TO TAKE (↑Z) EXIT
5364                                  ;CHECK FOR (↑U)
5365  023504  122702  000025          6$:     CMPB    #025,R2         ;SEE IF (↑U) TYPED
5366  023510  001006                          BNE     8$              ;BR IF NOT (↑U)
5367  023512  104401  012121                  TYPE    ,CNTRLU         ;ECHO (↑U)
5368  023516  062766  000004  000004  7$:     ADD     #4,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
5369  023524  000752                          BR      3$              ;BR TO TAKE (↑U) EXIT
5370                                  ;CHECK FOR (↑G)
5371  023526  122702  000007          8$:     CMPB    #007,R2         ;SEE IF (↑G) TYPED
5372  023532  001005                          BNE     9$              ;BR IF NOT (↑G)
5373  023534  104401  012126                  TYPE    ,CNTRLG         ;ECHO (↑G)
5374  023540  004737  021136                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
5375  023544  000764                          BR      7$              ;TAKE (↑U) RETURN
5376                                  ;CHECK FOR RUB-OUT (DELETE)
5377  023546  122702  000177          9$:     CMPB    #177,R2         ;SEE IF RUB-OUT (DEL) TYPED
5378  023552  001020                          BNE     14$             ;BR IF NOT RUB-OUT
5379  023554  005700                          TST     R0              ;CHECK THE CHARACTER COUNT
5380  023556  001726                          BEQ     2$              ;BR IF COUNT = 0
5381  023560  005701                          TST     R1              ;CHECK THE RUB-OUT INDICATOR
5382  023562  001003                          BNE     11$             ;BR IF WE HAD A PREVIOUS RUB-OUT
5383  023564  005201                          INC     R1              ;SET RUB-OUT INDICATOR
5384  023566  104401  012137                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH (\)
5385  023572  005037  005444          11$:    CLR     $CRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
5386  023576  005300                          DEC     R0              ;DECREMENT COUNT
5387  023600  116037  005212  005444          MOVB    BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
5388  023606  104401  005444                  TYPE    ,SCRACH         ;ECHO CHARACTER TO BE DELETED
5389  023612  000710                          BR      2$              ;GO READ ANOTHER CHARACTER
5390  023614  005701                  14$:    TST     R1              ;CHECK THE RUB-OUT INDICATOR
5391  023616  001403                          BEQ     16$             ;BR IF INDICATOR IS NOT SET
5392  023620  104401  012137                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH
5393  023624  005001                          CLR     R1              ;CLEAR THE RUB-OUT INDICATOR
5394                                  ;CHECK FOR CARRIAGE RETURN
5395  023626  122702  000015          16$:    CMPB    #015,R2         ;SEE IF <CR> TYPED
5396  023632  001426                          BEQ     19$             ;BR IF <CR>
5397                                  ;HANDLE POSSIBLE DIGIT
5398  023634  005037  005444                  CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
5399  023640  110237  005444                  MOVB    R2,SCRACH       ;GET THIS CHARACTER INTO BUFFER
5400  023644  104401  005444                  TYPE    ,SCRACH         ;ECHO THE CHARACTER TYPED
```

```
5401   023650   110260   005212                    MOVB    R2,BUFFO(R0)    ;PUT CHARACTER INTO BUFFER
5402   023654   005200                             INC     R0              ;INCREMENT CHARACTER COUNTER
5403   023656   022700   000120                    CMP     #80.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
5404   023662   001264                             BNE     2$              ;BR IF NOT TOO MANY
5405   023664   104401   001325                    TYPE    $CRLF           ;TYPE <CR> AND <LF>
5406   023670   112760   000000   005212           MOVB    #0,BUFFO(R0)    ;PUT TERMINATING NULL INTO BUFFER
5407   023676   104401   005212                    TYPE    ,BUFFO          ;ECHO INPUT STRING
5408   023702   104401   001324                    TYPE    ,$QUES          ;TYPE <?> <CR> <LF>
5409   023706   000703                             BR      7$              ;TAKE ERROR EXIT FROM RDCHRS
5410   023710   104401   001325           19$:     TYPE    $CRLF           ;TYPE <CR> <LF>
5411   023714   112760   000000   005212           MOVB    #0,BUFFO(R0)    ;PUT TERMINATING NULL INTO BUFFER
5412   023722   062766   000006   000004           ADD     #6,4(SP)        ;FIX UP RETURN ADDRESS
5413   023730   012602                   24$:     MOV     (SP)+,R2        ;RESTORE R2
5414   023732   012601                             MOV     (SP)+,R1        ;RESTORE R1
5415   023734   000207                             RTS     PC              ;SUBROUTINE EXIT
5416
5417
5418
5419                                    ;;*****************************************************************
5420                                    .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
5421                                    ;*THIS SUBROUTINE TYPES "DRIVE SER. NO.  XXX" (IN DECIMAL), WITH LEADING
5422                                    ;*ZEROS SUPPRESSED.
5423                                    ;;*****************************************************************
5424   023736   104407           DRVSER: SAVREG                  ;SAVE R0-R5
5425   023740   004737   021170                    JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
5426   023744   112765   000141   000001           MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
5427   023752   004737   030146                    JSR     PC,DRVCAL       ;READ STATUS OF THIS DRIVE
5428   023756   104401   011731                    TYPE    ,DRIV           ;TYPE "DRIVE"
5429   023762   104401   011747                    TYPE    ,SERNM          ;TYPE " SER. NO.  "
5430   023766   016501   000054                    MOV     P.A11(R5),R1    ;GET "A" STATUS BYTE 11
5431   023772   012704   043156                    MOV     #$OCTVL,R4      ;GET ADDR OF CHAR BUFFER
5432   023776   010446                             MOV     R4,-(SP)        ;STORE IT ON STACK FOR $SUPRS
5433   024000   012703   000003                    MOV     #3,R3           ;INIT CHAR COUNT
5434   024004   006101                             ROL     R1              ;INITIALIZE BIT POSITIONS
5435   024006   006101                             ROL     R1
5436   024010   006101                   4$:       ROL     R1              ;GET NEXT 4 BITS
5437   024012   006101                             ROL     R1
5438   024014   006101                             ROL     R1
5439   024016   006101                             ROL     R1
5440   024020   010100                             MOV     R1,R0           ;GET A WORKING COPY
5441   024022   042700   177760                    BIC     #177760,R0      ;CLEAR ALL BUT LOW 4 BITS
5442   024026   052700   000060                    BIS     #'0,R0          ;CONVERT A DIGIT TO ASCII
5443   024032   110024                             MOVB    R0,(R4)+        ;PUT ASCII DIGIT INTO CHAR BUFFER
5444   024034   005303                             DEC     R3              ;DECREMENT CHAR COUNT
5445   024036   001364                             BNE     4$              ;BR IF NOT 3 CHARS YET
5446   024040   105014                             CLRB    (R4)            ;INSERT NULL TERMINATOR
5447   024042   004737   043370                    JSR     PC,@#$SUPRS     ;TYPE DRIVE SER. NUMBER
5448   024046   104401   001325                    TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
5449   024052   104410                             RESREG                  ;RESTORE R0-R5
5450   024054   000207                             RTS     PC              ;RETURN
5451
5452
5453
5454                                    ;;*****************************************************************
5455                                    .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
5456                                    ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXXX" (IN OCTAL),
```

```
                                        ;*WITH LEADING ZEROS SUPPRESSED.
                                        ;*********************************************************************
024056 004737 021170            CRTSER: JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
024062 142765 000020 000007             BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
024070 105737 003134                    TSTB    FORMAT          ;CHECK THE ACTUAL FORMAT
024074 001402                           BEQ     4$              ;BR IF 22 SECTORS
024076 105265 000004                    INCB    P.SECT(R5)      ;IF 20 SECTORS, READ SECTOR 1
024102 112765 000121 000001     4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
024110 013765 024404 000002             MOV     LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)/1456(OCT)
024116 112765 000002 000005             MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
024124 012765 053342 000010             MOV     #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
024132 012765 177776 000012             MOV     #-2,P.WC(R5)    ;SET WORD COUNT TO READ 2 WORDS
024140 004737 030146                    JSR     PC,DRVCAL       ;READ SERIAL NO. IN BSF
024144 104401 011740                    TYPE    ,CART           ;TYPE "CART."
024150 104401 011747                    TYPE    ,SERNM          ;TYPE "SER. NO.  "
024154 012746 053342                    MOV     #RWBUF,-(SP)    ;GET POINTER FOR $DB2O
024160 004737 043054                    JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL
024164 004737 043370                    JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
024170 104401 001325                    TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
024174 000207                           RTS     PC              ;RETURN

                                        ;THIS ROUTINE ASKS THE OPERATOR FOR THE DRIVE TYOPE AND ALL FUTURE
                                        ;TESTS WILL BE PERFORMED ASSUMING ALL THE DRIVES ARE OF THAT TYPE

024176 104401 001325            GETTYP: TYPE    ,$CRLF
024202 104401 010025                    TYPE    ,ASKTYP         ;ASK DRIVE TYPE
024206 004737 023424                    JSR     PC,RDCHRS       ;READ INPUT STRING
024212 024176                           GETTYP                  ;CONT-C RETURN
024214 024176                           GETTYP                  ;CONT-Z RET
024216 024176                           GETTYP                  ;CONT-U RET
024220 005700                           TST     R0              ;SEE IF CHAR TYPED
024222 001005                           BNE     2$              ;BR IF NO
024224 104401 005212            1$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
024230 104401 001324                    TYPE    ,$QUES          ;TYPE <?>
024234 000760                           BR      GETTYP          ;ASK AGAIN

024236 022737 000066 005212     2$:     CMP     #'6,BUFFO       ;SEE IF 6 TYPED
024244 001025                           BNE     3$              ;BR IF NO
024246 105037 003133                    CLRB    TYPFMT          ;SETUP FOR RK05
024252 012737 000632 024404             MOV     #632,LSTCYL
024260 012737 000025 024406             MOV     #25,HOLD1
024266 012737 010025 024410             MOV     #10025,HOLD2
024274 012737 000100 024412             MOV     #100,HOLD3
024302 012737 000060 024414             MOV     #60,HOLD4
024310 012737 000040 024416             MOV     #40,HOLD5
024316 000207                           RTS     PC
024320 022737 000067 005212     3$:     CMP     #'7,BUFFO       ;SEE IF 7 TYPED
024326 001336                           BNE     1$              ;BR IF NO
024330 152737 000004 003133             BISB    #B.CDT,TYPFMT   ;SETUP FOR RK07
024336 012737 001456 024404             MOV     #1456,LSTCYL
024344 012737 002025 024406             MOV     #2025,HOLD1
024352 012737 012025 024410             MOV     #12025,HOLD2
024360 012737 000200 024412             MOV     #200,HOLD3
024366 012737 000160 024414             MOV     #160,HOLD4
024374 012737 000100 024416             MOV     #100,HOLD5
024402 000207                           RTS     PC
```

```
5513
5514  024404  000000                    LSTCYL: 0
5515  024406  000000                    HOLD1:  0
5516  024410  000000                    HOLD2:  0
5517  024412  000000                    HOLD3:  0                       ;CYL DIF
5518  024414  000000                    HOLD4:  0                       ;COMPAT START CYL
5519  024416  000000                    HOLD5:  0                       ;HALF CYL DIF
5520
5521
5522
5523
5524
5525
5526
5527                                    ;;*****************************************************************
5528                                    .SBTTL  STALL - STALL FOR ST UNIT STALL TIMES
5529                                    ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
5530                                    ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
5531                                    ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
5532                                    ;* CALL -   JSR     PC,STALL
5533                                    ;;*****************************************************************
5534
5535  024420  010046                    STALL:  MOV     R0,-(SP)        ;SAVE R0
5536  024422  010146                            MOV     R1,-(SP)        ;SAVE R1
5537  024424  032777  000400  154506            BIT     #BIT08,@SWR     ;APPLY RANDOM STALL ?
5538  024432  001407                            BEQ     1$              ;BR IF NOT RANDOM
5539  024434  004737  042252                    JSR     PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
5540  024440  013700  042352                    MOV     $LONUM,R0       ;GET IT INTO R0
5541  024444  006200                            ASR     R0              ;SCALE IT DOWN
5542  024446  006200                            ASR     R0
5543  024450  000403                            BR      2$              ;GO STALL WITH RANDOM NO.
5544  024452  013700  005432            1$:     MOV     STALLS,R0       ;GET REQUESTED NO. OF STALLS
5545  024456  001406                            BEQ     6$              ;RETURN IF NO STALL REQUIRED
5546  024460  012701  000016            2$:     MOV     #14.,R1         ;SET CONSTANT FOR 40 US
5547  024464  005301                    4$:     DEC     R1              ;INNER LOOP COUNTER
5548  024466  001376                            BNE     4$              ;INNER LOOP BR
5549  024470  005300                            DEC     R0              ;OUTER LOOP COUNTER
5550  024472  001372                            BNE     2$              ;OUTER LOOP BR
5551  024474  012601                    6$:     MOV     (SP)+,R1        ;RESTORE R1
5552  024476  012600                            MOV     (SP)+,R0        ;RESTORE R0
5553  024500  000207                            RTS     PC              ;RETURN
5554
5555
5556
5557                                    ;;*****************************************************************
5558                                    ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF THE DATA PATTERN
5559                                    ;*IN TABLEB INTO ALL 256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
5560                                    ;;*****************************************************************
5561  024502  104407                    LODSEC: SAVREG                  ;SAVE R0-R5
5562  024504  012701  053342                    MOV     #RWBUF,R1       ;GET ADRS OF R/W BUF INTO R1
5563  024510  012700  005766            2$:     MOV     #TABLEB,R0      ;GET DATA PATTERN ADDRESS
5564  024514  012703  000020                    MOV     #16.,R3         ;INIT PATTERN COUNTER
5565  024520  012021                    4$:     MOV     (R0)+,(R1)+     ;MOVE A PATTERN WORD
5566  024522  020127  054342                    CMP     R1,#RWBUF+512.  ;SEE IF DONE YET
5567  024526  103003                            BHIS    9$              ;BR IF DONE
5568  024530  005303                            DEC     R3              ;DECREMENT PATTERN COUNTER
```

```
5569   024532  001372                          BNE     4$              ;BR IF NOT RESTARTING PATTERN YET
5570   024534  000765                          BR      2$              ;BR TO RESTART PATTERN
5571   024536  104410                  9$:     RESREG                  ;RESTORE R0-R5
5572   024540  000207                          RTS     PC              ;RETURN
5573
5574
5575
5576
5577
5578
5579                                    ;*****************************************************************
5580                                    .SBTTL  FINADR - COMPUTE FINAL PACK ADDRESS
5581                                    ;*THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
5582                                    ;*COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
5583                                    ;*WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
5584                                    ;*P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
5585                                    ;*PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
5586                                    ;*THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
5587                                    ;*DATA MISCOMPARE.
5588                                    ;*****************************************************************
5589   024542  104407                  FINADR: SAVREG                  ;SAVE R0-R5
5590   024544  013746  005474                  MOV     LASTWC,-(SP)    ;STORE WORD COUNT
5591   024550  005416                          NEG     (SP)            ;MAKE IT POSITIVE
5592   024552  005046                  18$:    CLR     -(SP)           ;MAKE ROOM ON STACK
5593   024554  116616  000003                  MOVB    3(SP),(SP)      ;STORE NO. OF SECTORS TRANSFERRED
5594   024560  005066  000002                  CLR     2(SP)           ;CLEAR LOCATION ON STACK
5595   024564  116566  000004  000002          MOVB    P.SECT(R5),2(SP) ;STORE STARTING SECTOR
5596   024572  066616  000002                  ADD     2(SP),(SP)      ;DETERMINE FINAL SECTOR ADDRESS
5597   024576  005066  000002                  CLR     2(SP)           ;CLEAR NO. OF TRACKS TRANSFERRED
5598   024602  012700  000026                  MOV     #22.,R0         ;SET FOR 22 SECTORS
5599   024606  105737  003134                  TSTB    FORMAT          ;DETERMINE THE FORMAT
5600   024612  001402                          BEQ     19$             ;BR IF 22 SECTORS
5601   024614  012700  000024                  MOV     #20.,R0         ;SET FOR 20 SECTORS
5602   024620  020016                  19$:    CMP     R0,(SP)         ;CHECK FOR SECTOR OVERFLOW
5603   024622  101004                          BHI     20$             ;NO, CHECK IF SECTOR CORRECT
5604   024624  160016                          SUB     R0,(SP)         ;DECREMENT SECTOR COUNT BY 20 OR 22
5605   024626  005266  000002                  INC     2(SP)           ;INCREMENT TRACKS TRANSFERRED
5606   024632  000772                          BR      19$             ;CHECK FOR SECTOR OVERFLOW
5607   024634  112637  005473          20$:    MOVB    (SP)+,FINSEC    ;STORE FINAL SECTOR
5608   024640  005046                          CLR     -(SP)           ;MAKE ROOM FOR TRACKS TRANSFERRED
5609   024642  116516  000005                  MOVB    P.TRCK(R5),(SP) ;STORE STARTING TRACK
5610   024646  066616  000002                  ADD     2(SP),(SP)      ;DETERMINE FINAL TRACK ADDRESS
5611   024652  005066  000002                  CLR     2(SP)           ;CLEAR FINAL CYLINDER
5612   024656  122716  000003          21$:    CMPB    #3,(SP)         ;CHECK FOR TRACK OVERFLOW
5613   024662  101005                          BHI     22$             ;NO, CHECK FINAL TRACK
5614   024664  162716  000003                  SUB     #3,(SP)         ;DECREMENT TRACK COUNT BY 3
5615   024670  005266  000002                  INC     2(SP)           ;INCR CYL COUNT
5616   024674  000770                          BR      21$             ;CHECK FOR TRACK OVERFLOW
5617   024676  112637  005472          22$:    MOVB    (SP)+,FINTRK    ;STORE FINAL TRACK
5618   024702  066516  000002                  ADD     P.CYLN(R5),(SP) ;CALCULATE FINAL CYLINDER
5619   024706  011637  005470                  MOV     (SP),FINCYL     ;STORE FINAL  CYLINDER
5620   024712  005726                          TST     (SP)+           ;CLEAN OFF STACK
5621   024714  104410                          RESREG                  ;RESTORE R0-R5
5622   024716  000207                          RTS     PC              ;RETURN
5623
5624
```

```
                                ;;*****************************************************************
                                .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.
                                ;;*****************************************************************

024720  104407                  LODBUF: SAVREG                  ;SAVE R0-R5
024722  013702  005466                  MOV     WDSXFR,R2       ;GET NO. OF WORDS
024726  005737  005446                  TST     PATRN           ;SEE IF QUICK VERIFY DATA TEST DESIRED
024732  001007                          BNE     3$              ;BR IF NOT QUICK VERIFY
024734  012700  005566                  MOV     #PAT00,R0       ;SET DATA PATTERN STARTING ADDRESS
024740  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
024744  012701  053342                  MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
024750  000422                          BR      30$             ;PROCEED
024752  005000                  3$:     CLR     R0              ;INIT PATTERN NUMBER
024754  032701  000001          4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
024760  001003                          BNE     6$              ;BR IF THIS BIT IS SET
024762  005200                          INC     R0              ;INCREMENT PATTERN NO.
024764  006201                          ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
024766  000772                          BR      4$              ;BR TO CHECK NEXT BIT
024770  006300                  6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
024772  006300                          ASL     R0
024774  006300                          ASL     R0
024776  006300                          ASL     R0
025000  006300                          ASL     R0
025002  062700  005566                  ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
025006  012746  000020                  MOV     #16.,-(SP)      ;SET PATTERN WORD COUNT
025012  013701  005504                  MOV     PMA,R1          ;SET BUFFER ADDRESS
025016  010003                  30$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
025020  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
025022  012321                  34$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
025024  005302                          DEC     R2              ;DECREMENT WORD COUNTER
025026  001403                          BEQ     44$             ;BR IF ALL DONE
025030  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
025032  001373                          BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
025034  000770                          BR      30$             ;BR TO REPEAT THE PATTERN
025036  005726                  44$:    TST     (SP)+           ;POP THE STACK
025040  104410                          RESREG                  ;RESTORE R0-R5
025042  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.
```

```
5681                                    ;;*****************************************************************
5682  025044  104407                    CMPBUF: SAVREG                  ;SAVE R0-R5
5683  025046  112737  000040  051665            MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
5684  025054  012737  000007  053234            MOV     #7,DF25+2
5685  025062  013702  005466                    MOV     WDSXFR,R2       ;SET NO. OF WORDS
5686  025066  005037  005444                    CLR     SCRACH          ;CLEAR COMPARE ERROR COUNT
5687  025072  004737  031224                    JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
5688  025076  005737  005446                    TST     PATRN           ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
5689  025102  001007                            BNE     3$              ;BR IF NOT
5690  025104  012700  005566                    MOV     #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
5691  025110  012746  000400                    MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
5692  025114  012701  053342                    MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
5693  025120  000422                            BR      30$             ;PROCEED
5694  025122  005000                    3$:     CLR     R0              ;INIT PATTERN NO.
5695  025124  032701  000001            4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
5696  025130  001003                            BNE     6$              ;BR IF THIS BIT IS SET
5697  025132  005200                            INC     R0              ;INCREMENT PATTERN NUMBER
5698  025134  006201                            ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
5699  025136  000772                            BR      4$              ;BR TO CHECK NEXT BIT
5700  025140  006300                    6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
5701  025142  006300                            ASL     R0
5702  025144  006300                            ASL     R0
5703  025146  006300                            ASL     R0
5704  025150  006300                            ASL     R0
5705  025152  062700  005566                    ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
5706  025156  012746  000020                    MOV     #16.,-(SP)      ;PATTERN WORD COUNT
5707  025162  013701  005504                    MOV     PMA,R1          ;SET BUFFER ADDRESS
5708  025166  010003                    30$:    MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
5709  025170  011604                            MOV     (SP),R4         ;INIT PATTERN WORD COUNT
5710  025172  022321                    34$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
5711  025174  001002                            BNE     31$             ;BR IF COMPARE ERROR
5712  025176  000137  025536                    JMP     40$             ;JUMP IF DATA COMPARES OK
5713  025202  105037  051665            31$:    CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
5714  025206  012737  000005  053234            MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
5715                                    ;COMMON COMPARE ERROR HANDLER
5716  025214  010237  001202            35$:    MOV     R2,$REG10       ;GET WORD NO.
5717  025220  163737  005466  001202            SUB     WDSXFR,$REG10
5718  025226  013737  001202  005474            MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
5719  025234  004737  024542                    JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
5720  025240  104407                            SAVREG                  ;SAVE R0-R5
5721  025242  013700  005470                    MOV     FINCYL,R0       ;GET CYL
5722  025246  113701  005472                    MOVB    FINTRK,R1       ;GET TRACK
5723  025252  113702  005473                    MOVB    FINSEC,R2       ;GET SECTOR
5724  025256  004737  030042                    JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
5725  025262  104410                            RESREG                  ;RESTORE R0-R5
5726  025264  032737  001000  005424            BIT     #BADSEC,RECODE
5727  025272  001116                            BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
5728  025274  005737  005444                    TST     SCRACH          ;CHECK THE ERROR COUNT
5729  025300  001010                            BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
5730  025302  104034                    46$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
5731  025304  012737  177777  001174    48$:    MOV     #-1,$REG5       ;INIT CYL NO.
5732  025312  005037  001176                    CLR     $REG6
5733  025316  005037  001200                    CLR     $REG7
5734  025322  005237  005444            36$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
5735  025326  032777  000001  153604            BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
5736  025334  001004                            BNE     38$             ;BR TO REPORT ALL ERRORS
```

```
5737  025336  022737  000012  005444          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
5738  025344  002504                          BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
5739  025346  023737  001174  005470  38$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
5740  025354  001010                          BNE     42$             ;BR IF YES
5741  025356  123737  001176  005472          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
5742  025364  001004                          BNE     42$             ;BR IF YES
5743  025366  123737  001200  005473          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
5744  025374  001412                          BEQ     44$             ;BR IF SAME PACK ADDRESS
5745  025376  013737  005470  001174  42$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
5746  025404  113737  005472  001176          MOVB    FINTRK,$REG6
5747  025412  113737  005473  001200          MOVB    FINSEC,$REG7
5748  025420  104115                          ERROR   115             ;TYPE NEW PACK ADDRESS
5749  025422  005437  001202          44$:    NEG     $REG10          ;GET WORD NO.
5750  025426  016337  177776  001204          MOV     -2(R3),$REG11   ;GET GOOD DATA
5751  025434  016137  177776  001206          MOV     -2(R1),$REG12   ;GET BAD DATA
5752  025442  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
5753  025450  005037  001210                  CLR     $REG13
5754  025454  006137  001212                  ROL     $REG14
5755  025460  006137  001210                  ROL     $REG13
5756  025464  063737  005504  001212          ADD     PMA,$REG14
5757  025472  005537  001210                  ADC     $REG13
5758  025476  063737  005506  001210          ADD     PMA+2,$REG13
5759  025504  010137  001214                  MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
5760  025510  162737  000002  001214          SUB     #2,$REG15
5761  025516  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
5762  025520  032777  000100  153412          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
5763  025526  001013                          BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
5764  025530  005737  005446          50$:    TST     PATRN           ;SEE IF DEFAULT DATA TEST
5765  025534  001400                          BEQ     40$             ;BR IF YES
5766  025536  005302                  40$:    DEC     R2              ;DECREMENT WORD COUNTER
5767  025540  001406                          BEQ     54$             ;BR IF ALL DONE
5768  025542  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
5769  025544  001002                          BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
5770  025546  000137  025166                  JMP     30$             ;JUMP TO REPEAT THE PATTERN
5771  025552  000137  025172          53$:    JMP     34$
5772  025556                          54$:
5773  025556  005726                  56$:    TST     (SP)+           ;POP THE STACK
5774  025560  104410                          RESREG                  ;RESTORE R0-R5
5775  025562  000207                          RTS     PC              ;RETURN
5776
5777
5778
5779                                  ;;*****************************************************************
5780                                  ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
5781                                  ;*IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (↑Z)
5782                                  ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
5783                                  ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
5784                                  ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
5785                                  ;*NO ACTION IS TAKEN.
5786                                  ;*   CALL -   JSR     PC,CTLOUT
5787                                  ;*     OR -   CKEXIT
5788                                  ;*****************************************************************
5789  025564  122737  000001  003127  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
5790  025572  001075                          BNE     10$             ;BR IF NOT RUNNING TESTS
5791  025574  005737  005442                  TST     INTCHR          ;SEE IF ANY TTY INPUT
5792  025600  001472                          BEQ     10$             ;BR IF NO INPUT
```

```
5793  025602  105737  003126              TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
5794  025606  001441                      BEQ     12$             ;BR IF YES
5795  025610  122737  000003  005442      CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
5796  025616  001026                      BNE     4$              ;BR IF NOT (↑C)
5797  025620  012700  013002              MOV     #ALLSYS,R0      ;SET RETURN ADDR = ALLSYS
5798  025624                      2$:
5799  025624  000005                  6$: RESET                   ;RESET ALL DEVICES
5800  025626  005037  005442              CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
5801  025632  105037  003127              CLRB    TSTING          ;CLEAR TTY ESCAPE ALLOW FLAG
5802  025636  012737  031504  003056      MOV     #ERRHDL,@#ABNL  ;RESTORE ERROR HANDLER ADDRESS
5803  025644  012706  001100              MOV     #STACK,SP       ;RESET THE STACK
5804  025650  112765  000113  000001      MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
5805  025656  004737  030146              JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
5806  025662  105037  003143              CLRB    TSTTYP          ;CLEAR OFFSET TEST IDENTIFIER
5807  025666  005037  001102              CLR     $TSTNM          ;CLEAR THE TEST NO.
5808  025672  000110                      JMP     @R0             ;EXIT FROM TESTS
5809  025674  122737  000032  005442  4$: CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
5810  025702  001021                      BNE     7$              ;BR IF NOT (↑Z)
5811  025704  012700  013446              MOV     #ASKSYS,R0      ;SET RETURN ADDR = ASKSYS
5812  025710  000745                      BR      2$              ;TAKE EXIT
5813  025712  122737  000003  005442 12$: CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
5814  025720  001012                      BNE     7$              ;BR IF NOT
5815  025722  104401  011567              TYPE    ,HLTRQD         ;TYPE "HALT REQUESTED"
5816  025726  104401  011537              TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN RDY"
5817  025732  012700  033542              MOV     #HLTPRG,R0      ;SET HALT ADDRESS
5818  025736  112737  000061  003155      MOVB    #'1,PASSNO      ;RESTORE PASS NO. TO 1
5819  025744  000727                      BR      2$              ;TAKE EXIT
5820  025746  122737  000007  005442  7$: CMPB    #007,INTCHR     ;SEE IF (↑G) TYPED
5821  025754  001002                      BNE     8$              ;BR IF NOT (↑G)
5822  025756  004737  021136              JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
5823  025762  004737  020774          8$: JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
5824  025766  000207                  10$: RTS     PC              ;RETURN
5825
5826
5827
5828                                  ;;*****************************************************************
5829                                  ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO THE DISK.
5830                                  ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PACK
5831                                  ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
5832                                  ;*USED.
5833                                  ;;*****************************************************************
5834  025770  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
5835                                  ;LOAD THE R/W BUFFER WITH DATA
5836  025772  012700  053342                  MOV     #RWBUF,R0      ;BUFFER ADDRESS
5837  025776  012701  000400                  MOV     #400,R1        ;400(OCT) WORDS
5838  026002  010320                  4$:     MOV     R3,(R0)+       ;LOAD A BUFFER WORD
5839  026004  005301                          DEC     R1             ;DECR COUNTER
5840  026006  001375                          BNE     4$             ;BR IF NOT DONE YET
5841                                  ;SET UP PARAMETERS
5842  026010  012765  053342  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
5843  026016  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
5844                                  ;WRITE THE DATA
5845  026024  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
5846  026032  004737  030146                  JSR     PC,DRVCAL      ;WRITE THE DATA
5847  026036  104410                          RESREG                 ;RESTORE R0-R5
5848  026040  000207                          RTS     PC             ;RETURN
```

```
                                        ;;*****************************************************************
                                        ;*LDMEM1 - LOAD MEMORY WITH INCREASING NUMBERS, STARTING WITH 1.
                                        ;*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.
                                        ;;*****************************************************************
026042  104407                  LDMEM1: SAVREG                  ;SAVE R0-R5
026044  005004                          CLR     R4              ;R4=0 INDICATES LDMEM1
026046  012703  000001                  MOV     #1,R3           ;INIT DATA WORD TO 1
026052  000403                          BR      LDM0            ;PROCEED
026054  104407                  LDMEM2: SAVREG                  ;SAVE R0-R5
026056  012704  177777                  MOV     #-1,R4          ;R4=177777 INDICATES LDMEM2
026062  013702  005466          LDM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
026066  013701  005504                  MOV     PMA,R1          ;GET MEM ADRS
026072  010321                  6$:     MOV     R3,(R1)+        ;LOAD A WORD INTO BUFFER
026074  005704                          TST     R4              ;SEE WHICH DATA DESIRED
026076  001001                          BNE     8$              ;BR IF NOT INCREMENTING DATA
026100  005203                          INC     R3              ;INCREMENT THE DATA
026102                          8$:
026102  005302                  10$:    DEC     R2              ;DECREMENT COUNTER
026104  001372                          BNE     6$              ;BR IF NOT DONE LOADING YET
026106  104410                  12$:    RESREG                  ;RESTORE R0-R5
026110  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.
                                        ;;*****************************************************************
026112  104407                  CKMEM1: SAVREG                  ;SAVE R0-R5
026114  005004                          CLR     R4              ;R4=0 INDICATES CKMEM1
026116  012703  000001                  MOV     #1,R3           ;INIT DATA TO 1
026122  000403                          BR      CKM0            ;PROCEED
026124  104407                  CKMEM2: SAVREG                  ;SAVE R0-R5
026126  012704  177777                  MOV     #-1,R4          ;R4=177777 INDICATES CKMEM2
026132  013702  005466          CKM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
026136  013701  005504                  MOV     PMA,R1          ;GET MEM ADRS
026142  005037  005444                  CLR     SCRACH
026146  112737  000040  051665          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
026154  012737  000007  053234          MOV     #7,DF25+2
026162  004737  031224                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
026166  020321                  6$:     CMP     R3,(R1)+        ;COMPARE A DATA WORD
026170  001552                          BEQ     16$             ;BR IF DATA COMPARES OK
                                ;COMPARE ERROR HANDLER
026172  010237  001202                  MOV     R2,$REG10       ;GET WORD NO.
026176  163737  005466  001202          SUB     WDSXFR,$REG10
026204  013737  001202  005474          MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
026212  004737  024542                  JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
```

```
5905  026216  104407                          SAVREG                  ;SAVE R0-R5
5906  026220  013700  005470                  MOV     FINCYL,R0       ;GET CYL
5907  026224  113701  005472                  MOVB    FINTRK,R1       ;GET TRACK
5908  026230  113702  005473                  MOVB    FINSEC,R2       ;GET SECTOR
5909  026234  004737  030042                  JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
5910  026240  104410                          RESREG                  ;RESTORE R0-R5
5911  026242  032737  001000  005424          BIT     #BADSEC,RECODE
5912  026250  001122                          BNE     16$             ;BR IF LISTED- DON'T REPORT ERROR
5913  026252  105037  051665                  CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
5914  026256  012737  000005  053234          MOV     #5,DF25+2       ;ADJ. ERROR DATA WORD COUNT
5915  026264  005737  005444          9$:     TST     SCRACH          ;SEE IF FIRST ERROR IN THIS BLOCK
5916  026270  001010                          BNE     10$             ;BR IF NOT FIRST ERROR
5917  026272  104034                  17$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
5918  026274  012737  177777  001174  11$:    MOV     #-1,$REG5       ;INIT CYL NO.
5919  026302  005037  001176                  CLR     $REG6
5920  026306  005037  001200                  CLR     $REG7
5921  026312  005237  005444          10$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
5922  026316  032777  000001  152614          BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
5923  026324  001004                          BNE     12$             ;BR TO REPORT ALL ERRORS
5924  026326  022737  000012  005444          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
5925  026334  002477                          BLT     21$             ;BR IF ERROR LIMIT EXCEEDED
5926  026336  023737  001174  005470  12$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
5927  026344  001010                          BNE     14$             ;BR IF YES
5928  026346  123737  001176  005472          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
5929  026354  001004                          BNE     14$             ;BR IF YES
5930  026356  123737  001200  005473          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
5931  026364  001412                          BEQ     15$             ;BR IF SAME PACK ADDRESS
5932  026366  013737  005470  001174  14$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADR FOR PRINTOUT
5933  026374  113737  005472  001176          MOVB    FINTRK,$REG6
5934  026402  113737  005473  001200          MOVB    FINSEC,$REG7
5935  026410  104115                          ERROR   115             ;TYPE NEW PACK ADRS
5936  026412  005437  001202          15$:    NEG     $REG10          ;GET WORD NO.
5937  026416  010337  001204                  MOV     R3,$REG11       ;GOOD DATA
5938  026422  016137  177776  001206          MOV     -2(R1),$REG12   ;BAD DATA
5939  026430  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
5940  026436  005037  001210                  CLR     $REG13
5941  026442  006137  001212                  ROL     $REG14
5942  026446  006137  001210                  ROL     $REG13
5943  026452  063737  005504  001212          ADD     PMA,$REG14
5944  026460  005537  001210                  ADC     $REG13
5945  026464  063737  005506  001210          ADD     PMA+2,$REG13
5946  026472  010137  001214                  MOV     R1,$REG15       ;GET VIRT ADRS FOR PRINTOUT
5947  026476  162737  000002  001214          SUB     #2,$REG15
5948  026504  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA
5949  026506  032777  000100  152424          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
5950  026514  001007                          BNE     21$             ;BR IF JUST 1
5951                                  ;PROCEED WITH COMPARISONS
5952  026516  005704                  16$:    TST     R4              ;SEE WHICH DATA DESIRED
5953  026520  001001                          BNE     18$             ;BR IF NOT INCREASING DATA
5954  026522  005203                          INC     R3              ;INCREMENT THE DATA WORD
5955  026524                          18$:
5956  026524  005302                  20$:    DEC     R2              ;DECR COUNTER
5957  026526  001402                          BEQ     21$             ;BR IF DONE COMPARING
5958  026530  000137  026166                  JMP     6$              ;CONTINUE COMPARING WORDS
5959  026534                          21$:
5960  026534  104410                  22$:    RESREG                  ;RESTORE R0-R5
```

```
 5961  026536  000207                          RTS     PC              ;RETURN
 5962
 5963
 5964
 5965
 5966
 5967                                  ;;*****************************************************************
 5968                                  ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
 5969                                  ;*INTO BSSOFT.
 5970                                  ;*****************************************************************
 5971  026540                          REDBSF:
 5972  026540  004737  021170                  JSR     PC,INITSS       ;INIT THE S.S.
 5973  026544  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
 5974  026552  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
 5975  026560  013765  024404  000002          MOV     LSTCYL,P.CYLN(R5)       ;SET CYL = 632/1456
 5976  026566  112765  000002  000005          MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
 5977  026574  012703  000012                  MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
 5978  026600  105737  003134                  TSTB    FORMAT
 5979  026604  001403                          BEQ     6$              ;BR IF 22 SECTORS
 5980  026606  105265  000004                  INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
 5981  026612  005203                          INC     R3
 5982  026614  012765  177400  000012  6$:     MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
 5983  026622  012765  004162  000010          MOV     #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
 5984  026630  012737  027004  003056  8$:     MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
 5985  026636  105037  003145                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
 5986  026642  004737  030146                  JSR     PC,DRVCAL       ;READ THE FACTORY BSF
 5987  026646  105737  003145                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
 5988  026652  001413                          BEQ     12$             ;BR IF NOT
 5989  026654  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
 5990  026662  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
 5991  026666  001360                          BNE     8$              ;BR IF NOT YET
 5992  026670  004737  031224          10$:    JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
 5993  026674  104120                          ERROR   120             ;ABORTING- BAD BSF READ
 5994  026676  000137  033542                  JMP     HLTPRG          ;HALT- CAN'T PROCEED
 5995  026702  012765  003162  000010  12$:     MOV     #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
 5996  026710  110365  000004                  MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
 5997  026714  012703  000026                  MOV     #22.,R3         ;SET 22 SECTOR LIMIT
 5998  026720  105737  003134                  TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
 5999  026724  001402                          BEQ     14$             ;BR IF YES
 6000  026726  012703  000023                  MOV     #19.,R3         ;SET 20 SECTOR LIMIT
 6001  026732  012737  027004  003056  14$:     MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
 6002  026740  105037  003145                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
 6003  026744  004737  030146                  JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
 6004  026750  105737  003145                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
 6005  026754  001407                          BEQ     16$             ;BR IF NOT
 6006  026756  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
 6007  026764  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
 6008  026770  003760                          BLE     14$             ;BR IF NOT YET
 6009  026772  000736                          BR      10$             ;REPORT ERROR AND ABORT
 6010  026774  012737  031504  003056  16$:     MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
 6011  027002  000207                          RTS     PC              ;RETURN
 6012
 6013                                  ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
 6014                                  ;*POSSIBLE ERRORS IN READING BAD SECTORS.
 6015  027004  032765  130600  000034  BDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
 6016  027012  001002                          BNE     4$              ;BR IF A READ ERROR OCCURRED
```

```
 6017  027014  000137  031504                4$:     JMP     ERRHDL          ;GO HANDLE OTHER ERROR
 6018  027020  105237  003145                        INCB    WCEFLG          ;SET ERROR FLAG
 6019  027024  000137  033754                        JMP     RETNML          ;TAKE NORMAL DRIVER RETURN
 6020
 6021
 6022
 6023                                        ;;*****************************************************************
 6024                                        ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
 6025                                        ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
 6026                                        ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
 6027                                        ;*2 BSF'S.
 6028                                        ;;*****************************************************************
 6029  027030  104407                        TYPBSF: SAVREG                  ;SAVE R0-R5
 6030  027032  005004                                CLR     R4              ;INIT BSF FILE INDICATOR
 6031  027034  012700  004172                        MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
 6032  027040  104401  011762                        TYPE    ,FACTBS         ;TYPE "FACTORY"
 6033  027044  104401  012007                        TYPE    ,BDSECT         ;TYPE "BAD SECTORS"
 6034  027050  104401  051432                3$:     TYPE    ,DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
 6035  027054  104401  001325                        TYPE    ,$CRLF
 6036  027060  005001                                CLR     R1              ;INIT LIMIT COUNTER
 6037  027062  005710                        4$:     TST     (R0)            ;SEE IF A SECTOR LISTED HERE
 6038  027064  100007                                BPL     8$              ;BR IF YES
 6039  027066  005701                                TST     R1              ;SEE IF FILE IS EMPTY
 6040  027070  001032                                BNE     14$             ;BR IF NOT EMPTY
 6041  027072  104401  012027                        TYPE    ,NOFALS         ;TYPE "NONE"
 6042  027076  104401  001325                        TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 6043  027102  000425                                BR      14$
 6044  027104  012046                        8$:     MOV     (R0)+,-(SP)     ;GET A CYL NO.
 6045  027106  104402                                TYPOC                   ;TYPE IT
 6046  027110  104401  012147                        TYPE    ,SPACE2         ;TYPE SEPARATORS
 6047  027114  011003                                MOV     (R0),R3         ;GET TRACK AND SECTOR
 6048  027116  105003                                CLRB    R3
 6049  027120  000303                                SWAB    R3              ;GET THE TRACK NO.
 6050  027122  010346                                MOV     R3,-(SP)
 6051  027124  104402                                TYPOC                   ;TYPE IT
 6052  027126  104401  012147                        TYPE    ,SPACE2         ;TYPE SEPARATORS
 6053  027132  111003                                MOVB    (R0),R3         ;GET SECTOR NO.
 6054  027134  010346                                MOV     R3,-(SP)
 6055  027136  104402                                TYPOC                   ;TYPE IT
 6056  027140  104401  001325                        TYPE    ,$CRLF
 6057  027144  005720                                TST     (R0)+           ;INCR THE POINTER
 6058  027146  005201                                INC     R1              ;INCR THE COUNTER
 6059  027150  020127  000176                        CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
 6060  027154  002742                                BLT     4$              ;BR IF NOT YET
 6061  027156  005704                        14$:    TST     R4              ;SEE IF BOTH FILES TYPED YET
 6062  027160  001006                                BNE     18$             ;BR IF YES
 6063  027162  005204                                INC     R4              ;SET INDICATOR FOR SOFT. FILE
 6064  027164  012700  003172                        MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
 6065  027170  104401  011775                        TYPE    ,SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
 6066  027174  000725                                BR      3$              ;GO TYPE SOFT. BSF
 6067  027176  104410                        18$:    RESREG                  ;RESTORE R0-R5
 6068  027200  000207                                RTS     PC              ;RETURN
 6069
 6070
 6071
 6072                                        ;;*****************************************************************
```

```
                                        ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
                                        ;*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
                                        ;*************************************************************
027202  104407                  FINMEM: SAVREG                  ;SAVE R0-R5
027204  016500  000002                  MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
027210  116501  000005                  MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
027214  116502  000004                  MOVB    P.SECT(R5),R2   ;SECTOR NO.
027220  005003                          CLR     R3              ;INIT SECTOR COUNT TO 0
027222  026500  000030          6$:     CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
027226  001006                          BNE     8$              ;BR IF NOT EQUAL
027230  126501  000027                  CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
027234  001003                          BNE     8$              ;BR IF NOT EQUAL
027236  126502  000026                  CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
027242  001404                          BEQ     12$             ;BR IF ADRS ARE EQUAL
027244  005203                  8$:     INC     R3              ;INCR SECTOR COUNT
027246  004737  027266                  JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
027252  000763                          BR      6$              ;GO COMPARE ADDRESSES
027254  000303                  12$:    SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
027256  010337  005466                  MOV     R3,WDSXFR       ;STORE IT
027262  104410                          RESREG                  ;RESTORE R0-R5
027264  000207                          RTS     PC              ;RETURN

                                        ;*************************************************************
                                        ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
                                        ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
                                        ;*IN R2.
                                        ;*************************************************************
027266  012704  000025          INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
027272  105737  003134                  TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
027276  001402                          BEQ     4$              ;BR IF YES
027300  012704  000023                  MOV     #19.,R4         ;SET SECTOR LIMIT
027304  005202                  4$:     INC     R2              ;INCR. SECTOR NO.
027306  020204                          CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
027310  003407                          BLE     6$              ;BR IF NOT
027312  005002                          CLR     R2              ;SET SECTOR = 0
027314  005201                          INC     R1              ;INCR. TRACK NO.
027316  020127  000002                  CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
027322  003402                          BLE     6$              ;BR IF NOT
027324  005001                          CLR     R1              ;SET TRACK = 0
027326  005200                          INC     R0              ;INCR. CYLINDER NO.
027330  000207                  6$:     RTS     PC              ;RETURN


                                        ;*************************************************************
                                        ;*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
                                        ;*THIS SUBROUTINE UPDATES PMA, PMA+2, P.BALO(R5),
                                        ;*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
                                        ;*TERMINATED BY A BAD SECTOR ERROR (BSE), A DATA CHECK ERROR (DCK),
                                        ;*OR A HEADER VRC ERROR (HVRC).
                                        ;*************************************************************
027332  104407                  MIDXFR: SAVREG                  ;SAVE R0-R5
027334  004737  027202                  JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
027340  016500  000030                  MOV     P.DCYL(R5),R0   ;GET CYL NO.
```

```
6129  027344  116501  000027                  MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
6130  027350  116502  000026                  MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
6131  027354  013703  005466                  MOV     WDSXFR,R3       ;GET NO. OF WORDS TRANSFERRED
6132  027360  032737  000020  005424          BIT     #DCKERR,RECODE  ;SEE IF DATA CHECK ERROR OCCURRED
6133  027366  001004                          BNE     9$              ;BR IF YES
6134  027370  062703  000400                  ADD     #400,R3         ;SKIP BAD SECTOR
6135  027374  004737  027266                  JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
6136  027400  010065  000002          9$:     MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
6137  027404  110165  000005                  MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
6138  027410  110265  000004                  MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
6139  027414  060365  000012                  ADD     R3,P.WC(R5)     ;UPDATE P.WC(R5)
6140  027420  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;SEE IF BUS ADRS INCR INHIB SET
6141  027426  001022                          BNE     16$             ;BR IF YES
6142  027430  005004                          CLR     R4              ;GET BYTES XFERRED
6143  027432  006103                          ROL     R3
6144  027434  006104                          ROL     R4
6145  027436  060337  005504                  ADD     R3,PMA          ;UPDATE PMA,PMA+2
6146  027442  005537  005506                  ADC     PMA+2
6147  027446  060437  005506                  ADD     R4,PMA+2
6148  027452  013765  005504  000010          MOV     PMA,P.BALO(R5)  ;UPDATE P.BALO(R5)
6149  027460  013700  005506                  MOV     PMA+2,R0
6150  027464  042700  177774                  BIC     #177774,R0
6151  027470  150065  000007                  BISB    R0,P.BAHI(R5)   ;UPDATE P.BAHI(R5)
6152  027474  104410                  16$:    RESREG                  ;RESTORE R0-R5
6153  027476  000207                          RTS     PC              ;RETURN
6154
6155
6156
6157                                  ;;*****************************************************************
6158                                  ;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
6159                                  ;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
6160                                  ;;*****************************************************************
6161  027500  104407                  SVPRMS: SAVREG                  ;SAVE R0-R5
6162  027502  012700  002642                  MOV     #PARM0+2,R0     ;ADRS OF PARAMS
6163  027506  012701  005454                  MOV     #SAVPRS,R1      ;ADRS OF SAVE AREA
6164  027512  016537  000012  005466          MOV     P.WC(R5),WDSXFR  ;GET WORD COUNT
6165  027520  005437  005466                  NEG     WDSXFR          ;MAKE IT POSITIVE
6166  027524  012737  053342  005504          MOV     #RWBUF,PMA      ;INIT PMA TO RWBUF
6167  027532  005037  005506                  CLR     PMA+2           ;INIT PMA+2 TO 0
6168  027536  000405                          BR      GTO             ;CONTINUE
6169  027540  104407                  GTPRMS: SAVREG                  ;SAVE R0-R5
6170  027542  012700  005454                  MOV     #SAVPRS,R0      ;ADRS OF SAVE AREA
6171  027546  012701  002642                  MOV     #PARM0+2,R1     ;ADRS OF PARAMS
6172  027552  012702  000005          GTO:    MOV     #5,R2           ;SET FOR 5 WORDS
6173  027556  012021                  6$:     MOV     (R0)+,(R1)+     ;MOVE A WORD
6174  027560  005302                          DEC     R2              ;SEE IF DONE YET
6175  027562  001375                          BNE     6$              ;BR IF NOT YET
6176  027564  104410                          RESREG                  ;RESTORE R0-R5
6177  027566  000207                          RTS     PC              ;RETURN
6178
6179
6180
6181                                  ;;*****************************************************************
6182                                  ;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS
6183                                  ;*OR DATA CHECK ERRORS, IF ENCOUNTERED.
6184                                  ;*THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
```

```
                                        ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED,
                                        ;*AND IT IS ALSO USED TO HANDLE THE DATA CHECK ERRORS CAUSED
                                        ;*BY READING DATA WITH OFFSETS, IN THE OVERWRITE AND DATA COMPATIBILITY
                                        ;*TESTS.
                                        ;*********************************************************************
027570  010446                  TRNSFR: MOV     R4,-(SP)        ;SAVE R4 ON STACK
027572  005004                          CLR     R4              ;CLEAR ERROR INDICATOR
027574  105037  003144                  CLRB    DCEFLG          ;CLEAR DCK ERROR FLAG
027600  005037  005424          4$:     CLR     RECODE          ;CLEAR ERROR FLAGS
027604  012737  020556  003056          MOV     #DCKHDL,A.ABNL  ;SET DCK ERROR HANDLER ADRS
027612  105037  003135                  CLRB    ERRCNT          ;CLEAR ERROR RETRY COUNT
027616  004737  030146                  JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
027622  032737  000026  005424          BIT     #BSERR!DCKERR!HVRCER,RECODE  ;SEE IF BSE OR DCK OR HVRC ERROR OCCURRED
027630  001410                          BEQ     6$              ;BR IF NOT
027632  005204                          INC     R4              ;INCR ERROR INDICATOR
027634  004737  023220                  JSR     PC,CKSCOR       ;DECR. DRIVE SCORE, IF NECESSARY
027640  004737  027332                  JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
027644  005765  000012                  TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
027650  001353                          BNE     4$              ;BR IF NOT
027652  005704                  6$:     TST     R4              ;SEE IF ANY ERRORS OCCURRED
027654  001404                          BEQ     8$              ;BR IF NOT
027656  004737  027540                  JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
027662  004737  027500                  JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
027666  012604                  8$:     MOV     (SP)+,R4        ;RESTORE R4
027670  012737  031504  003056          MOV     #ERRHDL,A.ABNL  ;RESTORE NORMAL ERROR HANDLER ADRS
027676  000207                          RTS     PC              ;RETURN


                                        ;*********************************************************************
                                        .SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
                                        ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
                                        ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
                                        ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
                                        ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
                                        ;*
                                        ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
                                        ;*RETURNS TO THE CALLER.  THE FIRST ENTRY ON THE STACK WILL BE THE
                                        ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
                                        ;*********************************************************************
027700  010237  001266          SRHTBS: MOV     R2,$TMP2
027704  010337  001270                  MOV     R3,$TMP3
027710  012637  001272                  MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
027714  011402                          MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
027716  012437  001264                  MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
027722  062737  001000  001264          ADD     #1000,$TMP1
027730  005003                          CLR     R3              ;CLEAR R3 FOR COUNT
027732  062702  000010                  ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
027736  005712                  1$:     TST     (R2)            ;TEST IF ALL ONES
027740  100430                          BMI     5$              ;YES-DONE
027742  020012                          CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
027744  001406                          BEQ     3$              ;YES-GO CHECK TRACK
027746  062702  000004                  ADD     #4,R2           ;ELSE BUMP POINTER
027752  020237  001264                  CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
027756  002021                          BGE     5$              ;YES-EXIT
027760  000766                          BR      1$              ;LOOP
```

```
6241  027762  005722                  3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
6242  027764  011237  001302                  MOV     (R2),$TMP10     ;GET TRK/SEC WORD
6243  027770  042737  174377  001302          BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
6244  027776  123701  001303                  CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
6245  030002  001402                          BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
6246  030004  005722                          TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
6247  030006  000753                          BR      1$              ;GO TEST NEXT CYL
6248  030010  005203                  4$:     INC     R3              ;BUMP BAD SECTOR COUNT
6249  030012  012246                          MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
6250  030014  042716  177700                  BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
6251  030020  000746                          BR      1$              ;GO CHECK REST OF FILE
6252  030022  010346                  5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
6253  030024  013702  001266                  MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
6254  030030  013703  001270                  MOV     $TMP3,R3        ;REGISTERS
6255  030034  013746  001272                  MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
6256  030040  000204                          RTS     R4              ;RETURN
6257                                  ;;**********************************************************************
6258                                  .SBTTL  BAD SECTOR CHECK ROUTINE
6259                                  ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
6260                                  ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
6261                                  ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
6262                                  ;*IS NOT LISTED THE FLAG IS RESET.
6263                                  ;;**********************************************************************
6264  030042  104407                  BDSRCK: SAVREG
6265  030044  012737  004162  030056          MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
6266  030052  004437  027700          2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
6267  030056  000000                  1$:     .WORD                   ;TABLE ADDRESS GOES HERE
6268  030060  012603                          MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
6269  030062  001015                          BNE     6$              ;IF ANY, GO TEST WHICH ONES
6270  030064  023727  030056  003162  7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
6271  030072  001404                          BEQ     3$              ;IF YES-EXIT
6272  030074  012737  003162  030056          MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
6273  030102  000763                          BR      2$              ;GO SEARCH IT
6274  030104  042737  001000  005424  3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
6275  030112  104410                  4$:     RESREG
6276  030114  000207                          RTS     PC              ;RETURN
6277  030116  022602                  6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
6278  030120  001403                          BEQ     8$              ;YES-GO SET BIT & EXIT
6279  030122  005303                          DEC     R3              ;DECREMENT BAD SECTOR COUNT
6280  030124  001374                          BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
6281  030126  000756                          BR      7$              ;ELSE GO SEARCH OTHER TABLE
6282  030130  052737  001000  005424  8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
6283  030136  005303                  9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
6284  030140  001764                          BEQ     4$              ;NUMBER
6285  030142  005726                          TST     (SP)+
6286  030144  000774                          BR      9$              ;EXIT
6287
6288
6289
6290
6291                                  ;;**********************************************************************
6292                                  .SBTTL  CALL DRIVER ROUTINE
6293                                  ;*ENTRY JSR PC,DRVCAL
6294                                  ;*                      WITH R5 POINTING TO PARAMETER BLOCK
6295                                  ;*RETURN RTS PC
6296                                  ;*
```

```
  6297                                  ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
  6298                                  ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
  6299                                  ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
  6300                                  ;*BLOCK WHEN THE ROUTINE IS CALLED.
  6301                                  ;*
  6302                                  ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
  6303                                  ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
  6304                                  ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
  6305                                  ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
  6306                                  ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
  6307                                  ;;*****************************************************************
  6308   030146                         DRVCAL:
  6309   030146  004737  024420                 JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
  6310   030152  004737  025564                 JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
  6311   030156  105037  003132                 CLRB    DONE            ;CLEAR DONE FLAG
  6312   030162  105037  003141                 CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
  6313   030166  004737  030220                 JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
  6314   030172  010537  030202                 MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
  6315   030176  004737  040222                 JSR     PC,C.INIT       ;CALL DRIVER
  6316   030202  000000                 4$:     .WORD                   ;P.B. ADDRESS GOES HERE
  6317   030204  004737  034572         6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
  6318   030210  105737  003132                 TSTB    DONE            ;DONE SET?
  6319   030214  001773                         BEQ     6$              ;NO-LOOP
  6320   030216  000207                         RTS     PC              ;YES-RETURN
  6321
  6322
  6323
  6324                                  ;;*****************************************************************
  6325                                  ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
  6326                                  ;;*****************************************************************
  6327   030220  104407                 STRCMD: SAVREG                  ;SAVE R0-R5
  6328   030222  012701  005162                 MOV     #PRVCMD,R1      ;ADDR OF PREV COMMAND STORAGE
  6329   030226  012700  005176                 MOV     #COMSTR,R0      ;ADDR OF CURRENT COMMAND STORAGE
  6330                                 ;STORE PREVIOUS COMMAND
  6331   030232  012021                         MOV     (R0)+,(R1)+
  6332   030234  012021                         MOV     (R0)+,(R1)+
  6333   030236  012021                         MOV     (R0)+,(R1)+
  6334   030240  012021                         MOV     (R0)+,(R1)+
  6335   030242  012021                         MOV     (R0)+,(R1)+
  6336   030244  012021                         MOV     (R0)+,(R1)+
  6337                                 ;STORE CURRENT COMMAND
  6338   030246  012701  005176         4$:     MOV     #COMSTR,R1
  6339   030252  012521                         MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BLK
  6340   030254  012521                         MOV     (R5)+,(R1)+
  6341   030256  012521                         MOV     (R5)+,(R1)+
  6342   030260  012521                         MOV     (R5)+,(R1)+
  6343   030262  012521                         MOV     (R5)+,(R1)+
  6344   030264  012521                         MOV     (R5)+,(R1)+
  6345   030266  104410                         RESREG                  ;RESTORE R0-R5
  6346   030270  000207                         RTS     PC              ;RETURN
  6347
  6348
  6349
  6350                                  ;;*****************************************************************
  6351                                  .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
  6352                                  ;*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.
                                        ;;*****************************************************************
030272  152737  000377  003132  ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
030300  032737  100000  005424          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
030306  001403                        BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
030310  105037  003135                CLRB    ERRCNT          ;CLEAR ERROR COUNT
030314  000413                        BR      1$              ;EXIT
030316  105737  003135          2$:   TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
030322  001410                        BEQ     1$              ;NO - SKIP TO EXIT
030324  005037  001174                CLR     $REG5
030330  113737  003135  001174        MOVB    ERRCNT,$REG5    ;GET RETRY COUNT
030336  104101                        ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
030340  105037  003135                CLRB    ERRCNT          ;CLEAR ERROR COUNT
030344  005037  005424          1$:   CLR     RECODE          ;CLEAR RECOVERY FLAGS
030350  000207                        RTS     PC              ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  TYPE ERROR ROUTINE
                                        ;*ENTRY -       JSR     PC,TYPERR
                                        ;*RETURN -      RTS     PC
                                        ;*
                                        ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                        ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
                                        ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
                                        ;*THE ERROR.
                                        ;;*****************************************************************
030352  104407                  TYPERR: SAVREG
030354  105237  003137                INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
030360  032777  020000  150552  9$:    BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
030366  001402                        BEQ     6$              ;BR IF NO
030370  000137  030774                JMP     20$
030374  005000                  6$:   CLR     R0              ;CLR R0 FOR ERROR NUMBER
030376  005005                        CLR     R5              ;INIT INDENT INDICATOR
030400  005105                        COM     R5
030402  113700  001114                MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
030406  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
030410  006300                        ASL     R0
030412  006300                        ASL     R0
030414  006300                        ASL     R0
030416  062700  001410          1$:   ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
030422  012037  030442                MOV     (R0)+,2$        ;GET EM POINTER
030426  001406                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
030430  104401  012133                TYPE    ,CR2LF
030434  104401  045412                TYPE    ,AS2SP2         ;TYPE "**  "
030440  104401                        TYPE                    ;TYPE ERROR MESSAGE (EM)
030442  000000                  2$:   .WORD   0               ;EM POINTER GOES HERE
030444  012037  030620          3$:   MOV     (R0)+,4$        ;GET DH POINTER
030450  001467                        BEQ     5$              ;BR IF THERE ISN'T ONE
030452  104401  001325                TYPE    ,$CRLF
030456  104401  045400                TYPE    ,TSTMSG         ;TYPE "    TEST "
030462  013746  001102                MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
030466  104403                        TYPOS                   ;TYPE IT
030470     002                        .BYTE   2               ;2 DIGITS
030471     000                        .BYTE   0               ;SUPPRESS LEADING ZEROS
030472  104401  012133                TYPE    ,CR2LF
```

```
6409  030476  032777  010000  150434          BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
6410  030504  001133                          BNE     20$             ;BR IF YES
6411  030506  104401  050534                  TYPE    ,DH105          ;TYPE "PREVIOUS COMMAND :"
6412  030512  104401  001325                  TYPE    ,$CRLF
6413  030516  104401  050726                  TYPE    ,DH101+10       ;TYPE PREV COMMAND HEADER
6414  030522  104401  001325                  TYPE    ,$CRLF
6415  030526  012701  000006                  MOV     #6,R1           ;SIX COMMAND VALUES
6416  030532  012702  001236                  MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
6417  030536  012246                  30$:    MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
6418  030540  104402                          TYPOC                   ;TYPE IT
6419  030542  104401  012147                  TYPE    ,SPACE2         ;TYPE SEPARATORS
6420  030546  005301                          DEC     R1              ;SEE IF 7 VALUES TYPED YET
6421  030550  001372                          BNE     30$             ;BR IF NOT
6422  030552  104401  001325                  TYPE    ,$CRLF
6423  030556  104401  012147                  TYPE    ,SPACE2         ;INDENT
6424  030562  104401  051005                  TYPE    ,DH102          ;TYPE HDR FOR BA DATA
6425  030566  104401  001325                  TYPE    ,$CRLF
6426  030572  104401  012147                  TYPE    ,SPACE2         ;INDENT
6427  030576  012246                          MOV     (R2)+,-(SP)
6428  030600  104402                          TYPOC                   ;TYPE PREV. HI BA BITS
6429  030602  104401  012147                  TYPE    ,SPACE2
6430  030606  011246                          MOV     (R2),-(SP)
6431  030610  104402                          TYPOC                   ;TYPE PREV. LO BA BITS
6432  030612  104401  012133                  TYPE    ,CR2LF
6433  030616  104401                          TYPE                    ;TYPE DATA HEADER
6434  030620  000000                  4$:     .WORD   0               ;DH POINTER GOES HERE
6435  030622  104401  001325                  TYPE    ,$CRLF
6436  030626  005005                          CLR     R5              ;INIT INDENT INDICATOR
6437  030630  032777  010000  150302  5$:     BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
6438  030636  001056                          BNE     20$             ;BR IF YES
6439  030640  012001                          MOV     (R0)+,R1        ;GET DT POINTER
6440  030642  001454                          BEQ     20$             ;BRANCH IF THERE ARE NONE
6441  030644  012000                          MOV     (R0)+,R0        ;GET DF POINTER
6442  030646  012002                          MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
6443  030650  112003                  10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
6444  030652  105720                          TSTB    (R0)+           ;BUMP PAST FORMAT WORD
6445  030654  005703                          TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
6446  030656  001417                          BEQ     14$             ;NO - SKIP DATA PRINT
6447  030660  005705                          TST     R5              ;SEE IF SHOULD INDENT
6448  030662  001002                          BNE     11$             ;BR IF NOT
6449  030664  104401  012147                  TYPE    ,SPACE2         ;INDENT
6450  030670  013146                  11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
6451  030672  104402                          TYPOC                   ;TYPE IT
6452  030674  005303                          DEC     R3              ;MORE DATA WORDS
6453  030676  001403                          BEQ     12$             ;NO-BRANCH
6454  030700  104401  012147                  TYPE    ,SPACE2         ;TYPE SEPARATORS
6455  030704  000771                          BR      11$             ;LOOP
6456  030706  005702                  12$:    TST     R2              ;SEE IF <CR>,<LF> NEEDED
6457  030710  001402                          BEQ     14$             ;BR IF NOT
6458  030712  104401  001325                  TYPE    ,$CRLF          ;TYPE IT
6459  030716  005302                  14$:    DEC     R2              ;MORE DH'S?
6460  030720  003425                          BLE     20$             ;NO-BRANCH
6461  030722  012037  030754          15$:    MOV     (R0)+,16$       ;GET NEXT DH POINTER
6462  030726  105710                          TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
6463  030730  001004                          BNE     34$             ;BR IF YES
6464  030732  104401  001325                  TYPE    ,$CRLF          ;SKIP EXTRA LINE
```

```
                                TYPE ERROR ROUTINE

 6465  030736  005005                        CLR     R5              ;RE-INIT INDENT INDICATOR
 6466  030740  000404                        BR      36$
 6467  030742  005105                34$:    COM     R5              ;COMPLEMENT INDENT INDICATOR
 6468  030744  001002                        BNE     36$             ;BR IF NO INDENT REQUIRED
 6469  030746  104401  012147                TYPE    ,SPACE2         ;INDENT
 6470  030752  104401                36$:    TYPE                    ;TYPE DH
 6471  030754  000000                16$:    .WORD   0               ;DH POINTER GOES HERE
 6472  030756  104401  001325                TYPE    ,$CRLF
 6473  030762  105710                        TSTB    (R0)            ;TYPE A DT?
 6474  030764  001331                        BNE     10$             ;YES-BRANCH
 6475  030766  062700  000002                ADD     #2,R0           ;INCREMENT DF POINTER
 6476  030772  000751                        BR      14$             ;SEE IF END OF DF BLOCK
 6477  030774  104410                20$:    RESREG
 6478  030776  000207                        RTS     PC
 6479                                ;;*****************************************************************
 6480                                .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
 6481                                ;*ENTRY:        JSR PC, CONERR
 6482                                ;*RETURN:       RTS PC'
 6483                                ;*
 6484                                ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
 6485                                ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
 6486                                ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
 6487                                ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
 6488                                ;*AT THIS TIME.
 6489                                ;;*****************************************************************
 6490  031000  104407                CONERR: SAVREG                  ;SAVE R0-R5
 6491  031002  152737  000377  003132        @ISB    #377,DONE       ;SET DONE FLAG
 6492  031010  105237  003135                INCB    ERRCNT          ;INCREMENT ERROR COUNT
 6493  031014  004737  033764                JSR     PC,TOPROC       ;LOAD RK REGS INTO $REGS
 6494  031020  032737  000001  003062        BIT     #BIT0,E.CONT    ;ERROR 0?
 6495  031026  001402                        BEQ     1$              ;NO-BRANCH
 6496  031030  104064                        ERROR   64              ;CLEAR CONT DID NOT CLEAR ERROR
 6497  031032  000470                        BR      7$
 6498  031034  032737  000002  003062  1$:   BIT     #BIT1,E.CONT    ;ERROR 1?
 6499  031042  001402                        BEQ     2$              ;NO-BRANCH
 6500  031044  104065                        ERROR   65              ;NO ATTENTION IN ATTENTION SUM REG
 6501  031046  000462                        BR      7$
 6502  031050  032737  000004  003062  2$:   BIT     #BIT2,E.CONT    ;ERROR 2?
 6503  031056  001407                        BEQ     3$              ;NO-BRANCH
 6504  031060  105737  003141                TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
 6505  031064  001402                        BEQ     15$             ;BR IF NOT
 6506  031066  105237  003142                INCB    REISSU          ;SET FLAG TO RE-ISSUE COMMAND
 6507  031072  104066                15$:    ERROR   66              ;UNSOLICITED ATTENTION
 6508  031074  000447                        BR      7$
 6509  031076  032737  000010  003062  3$:   BIT     #BIT3,E.CONT    ;ERROR 3?
 6510  031104  001402                        BEQ     4$              ;NO-BRANCH
 6511  031106  104067                        ERROR   67              ;UNEXPECTED DATA TYPE ERROR
 6512  031110  000441                        BR      7$
 6513  031112  032737  000020  003062  4$:   BIT     #BIT4,E.CONT    ;ERROR 4?
 6514  031120  001402                        BEQ     5$              ;NO-BRANCH
 6515  031122  104070                        ERROR   70              ;ATTENTION DID NOT RESET WITH CLEAR
 6516  031124  000433                        BR      7$
 6517  031126  032737  000040  003062  5$:   BIT     #BIT5,E.CONT    ;ERROR 5?
 6518  031134  001402                        BEQ     6$              ;NO-BRANCH
 6519  031136  104071                        ERROR   71              ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
 6520  031140  000425                        BR      7$
```

```
6521  031142  032737  000400  003062  6$:     BIT     #BIT8,E.CONT
6522  031150  001401                          BEQ     8$
6523  031152  104072                          ERROR   72              ;DATA LATE WHEN UNLOADING HEADER
6524  031154  032737  001000  003062  8$:     BIT     #BIT9,E.CONT
6525  031162  001401                          BEQ     9$
6526  031164  104073                          ERROR   73              ;CONTROLLER ERROR DURING DRIVER SERVICE
6527  031166  032737  002000  003062  9$:     BIT     #BIT10,E.CONT
6528  031174  001401                          BEQ     10$
6529  031176  104074                          ERROR   74              ;DRIVE DETECTED PARITY ERROR
6530  031200  032737  100000  003062  10$:    BIT     #BIT15,E.CONT
6531  031206  001401                          BEQ     11$
6532  031210  104052                          ERROR   52              ;MULTIPLE DRIVE SELECT
6533  031212  104075                  11$:    ERROR   75              ;UNDEFINED ERROR
6534  031214  005037  003062          7$:     CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
6535  031220  000137  033560                  JMP     BGNRTY          ;GO DO RETRY
6536                                  ;;*****************************************************************
6537                                  .SBTTL  REPORT SUPPORT ROUTINE
6538                                  ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
6539                                  ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING.  ALL THESE MAY
6540                                  ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
6541  031224                          REPSUP:
6542  031224  104407                          SAVREG
6543  031226  005037  005426                  CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
6544  031232  116537  000001  005426          MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
6545  031240  012700  001162                  MOV     #$REG0,R0       ;FOR REPORTING
6546  031244  016520  000002                  MOV     P.CYLN(R5),(R0)+
6547  031250  116520  000005                  MOVB    P.TRCK(R5),(R0)+
6548  031254  105020                          CLRB    (R0)+
6549  031256  116520  000004                  MOVB    P.SECT(R5),(R0)+
6550  031262  105020                          CLRB    (R0)+
6551  031264  016520  000012                  MOV     P.WC(R5),(R0)+
6552  031270  012700  001174                  MOV     #$REG5,R0
6553  031274  116503  000007                  MOVB    P.BAHI(R5),R3
6554  031300  042703  177774                  BIC     #177774,R3
6555  031304  010337  001256                  MOV     R3,$REG36       ;HI BA BITS
6556  031310  016537  000010  001260          MOV     P.BALO(R5),$REG37 ;LO BA BITS
6557  031316  016520  000016                  MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
6558  031322  016520  000020                  MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
6559  031326  016520  000030                  MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
6560  031332  016520  000026                  MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
6561  031336  016520  000022                  MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
6562                                                                  ;FOR ALL REPORTS (TO BE
6563  031342  016520  000024                  MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
6564  031346  016520  000032                  MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
6565  031352  016520  000036                  MOV     P.DS(R5),(R0)+
6566  031356  016520  000034                  MOV     P.ER(R5),(R0)+
6567  031362  016520  000040                  MOV     P.AOD(R5),(R0)+
6568  031366  016520  000042                  MOV     P.BOD(R5),(R0)+
6569  031372  016520  000044                  MOV     P.AD1(R5),(R0)+
6570  031376  016520  000046                  MOV     P.BD1(R5),(R0)+
6571  031402  016520  000050                  MOV     P.A10(R5),(R0)+
6572  031406  016520  000052                  MOV     P.B10(R5),(R0)+
6573  031412  016520  000054                  MOV     P.A11(R5),(R0)+
6574  031416  016520  000056                  MOV     P.B11(R5),(R0)+
6575                                  ;STORE PREVIOUS COMMAND FOR PRINTOUT
6576  031422  012701  001236                  MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
```

```
6577  031426  012700  005162                    MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
6578  031432  112021                            MOVB    (R0)+,(R1)+     ;DRIVE NO.
6579  031434  105021                            CLRB    (R1)+
6580  031436  112021                            MOVB    (R0)+,(R1)+     ;COMMAND
6581  031440  105021                            CLRB    (R1)+
6582  031442  012021                            MOV     (R0)+,(R1)+     ;CYL ADDRESS
6583  031444  116021  000001                    MOVB    1(R0),(R1)+     ;TRACK
6584  031450  105021                            CLRB    (R1)+
6585  031452  111021                            MOVB    (R0),(R1)+      ;SECTOR
6586  031454  105021                            CLRB    (R1)+
6587  031456  016021  000006                    MOV     6(R0),(R1)+     ;WORD COUNT
6588  031462  116003  000003                    MOVB    3(R0),R3        ;HI BA BITS
6589  031466  042703  177774                    BIC     #177774,R3
6590  031472  010321                            MOV     R3,(R1)+
6591  031474  016011  000004                    MOV     4(R0),(R1)      ;LO BA BITS
6592  031500  104410                            RESREG
6593  031502  000207                            RTS     PC
6594                                ;;*****************************************************************
6595                                .SBTTL  REPORT ERROR ROUTINE
6596                                ;* ENTRY           JSR     PC, ERRHDL
6597                                ;*RETURN         RTS     PC
6598                                ;*
6599                                ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
6600                                ;*IN THE OPERATION.  THE ROUTINE DETERMINES WHICH COMMAND WAS
6601                                ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
6602                                ;;*****************************************************************
6603
6604  031504  104407                ERRHDL: SAVREG
6605  031506  152737  000377  003132            BISB    #377,DONE       ;SET DONE FLAG
6606  031514  105237  003135                    INCB    ERRCNT          ;INCREMENT ERROR COUNT
6607  031520  005037  005424                    CLR     RECODE          ;CLEAR RECOVERY CODE WORD
6608  031524  032737  000400  005424  ER2ENT:   BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
6609  031532  001402                            BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
6610  031534  012705  002724                    MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
6611  031540  012737  033736  003056  52$:      MOV     #RETANL,A.ABNL  ;SET NOW ABNORMAL AND NORMAL RETURN FOR
6612  031546  012737  033754  003054            MOV     #RETNML,A.NORM  ;DRIVER OPERATIONS IN ERROR PROCESSING
6613  031554  004737  031224                    JSR     PC,REPSUP       ;GO SET UP REGISTERS FOR REPORT
6614                                                                    ;NOW BEGIN TESTING THE ERROR
6615                                                                    ;BITS. THE SEQUENCE IN
6616                                                                    ;WHICH THEY ARE TESTED IS
6617                                                                    ;CONSIDERED SIGNIFICANT IN
6618                                                                    ;THAT ERRORS OF A MORE
6619                                                                    ;CATASTROPHIC NATURE ARE FIRST
6620                                                                    ;TESTED.
6621                                                                    ;
6622                                                                    ;IF AN ERROR IS FOUND SET,
6623                                                                    ;THAT ERROR IS REPORTED AND
6624                                                                    ;THE REPORTING IS TERMINATED.
6625                                                                    ;IF ADDITIONAL ERRORS ARE SET,
6626                                                                    ;THE RK611 REGISTER PRINTOUTS
6627                                                                    ;WILL SHOW THIS BUT THE
6628                                                                    ;REGISTER CONTENTS MUST BE
6629                                                                    ;MANUALLY DECODED TO LOCATE THE
6630                                                                    ;SECOND ERROR
6631  031560  016504  000034                    MOV     P.ER(R5),R4     ;SET R4 TO ERROR REGISTER
6632  031564  032765  020000  000020            BIT     #UPE,P.CS2(R5)  ;TEST UPE. IF UES-SET
```

```
6633  031572  001406                          BEQ     1$                  ;REPORT ERROR
6634  031574  104001                          ERROR   1
6635  031576  052737  000200  005424          BIS     #ABORT,RECODE
6636  031604  000137  033236                  JMP     37$
6637  031610  032765  004000  000020  1$:     BIT     #NEM,P.CS2(R5)      ;TEST NON-EXISTANT MEMORY
6638  031616  001406                          BEQ     2$
6639  031620  104002                          ERROR   2
6640  031622  052737  000200  005424          BIS     #ABORT,RECODE
6641  031630  000137  033236                  JMP     37$
6642  031634  032765  010000  000020  2$:     BIT     #NED,P.CS2(R5)      ;TEST NON-EXISTANT DRIVE
6643  031642  001412                          BEQ     3$
6644  031644  032765  000400  000020          BIT     #UFE,P.CS2(R5)      ;TEST IF NED & UFE BOTH SET
6645  031652  001403                          BEQ     38$
6646  031654  104035                          ERROR   35                  ;NED & UFE BOTH SET ERROR
6647  031656  000137  033236                  JMP     37$
6648  031662  104003          38$:            ERROR   3                   ;NED ONLY
6649  031664  000137  033236                  JMP     37$
6650  031670  032765  000400  000020  3$:     BIT     #UFE,P.CS2(R5)      ;TEST UNIT FIELD ERROR
6651  031676  001412                          BEQ     4$
6652  031700  032765  010000  000020          BIT     #NED,P.CS2(R5)      ;TEST IF UFE & NED BOTH SET
6653  031706  001403                          BEQ     39$                 ;NO-SKIP
6654  031710  104035                          ERROR   35                  ;REPORT NED & UFE BOTH SET
6655  031712  000137  033236                  JMP     37$
6656  031716  104004          39$:            ERROR   4                   ;REPORT UFE ONLY
6657  031720  000137  033236                  JMP     37$
6658  031724  032765  000100  000014  4$:     BIT     #CMDTO,P.PRST(R5) ;
6659  031732  001423                          BEQ     5$
6660  031734  004737  033764                  JSR     PC,TOPROC           ;GO PROCESS TIMEOUT
6661  031740  032737  002000  005424          BIT     #TWOTOS,RECODE      ;2ND TIMEOUT IN TIMEOUT PROC?
6662  031746  001007                          BNE     40$                 ;YES - SKIP TO ERROR 56
6663  031750  032737  000400  005424          BIT     #LEV2ER,RECODE      ;TEST IF LEVEL 2 ERROR
6664  031756  001006                          BNE     41$                 ;YES - SKIP TO ERROR 57
6665  031760  104005                          ERROR   5                   ;ELSE MAKE FULL TIMEOUT REPORT
6666  031762  000137  033236                  JMP     37$
6667  031766  104056          40$:            ERROR   56                  ;TWO TIMEOUTS ERROR REPORT
6668  031770  000137  033236                  JMP     37$
6669  031774  104057          41$:            ERROR   57                  ;2ND LEVEL ERROR REPORT
6670  031776  000137  031524                  JMP     ER2ENT              ;GO BUILD AND MAKE 2ND REPORT
6671  032002  032765  010000  000014  5$:     BIT     #DRVSZD,P.PRST(R5) ;SEE IF DRIVE SIEZED BY OTHER PORT
6672  032010  001403                          BEQ     65$                 ;BR IF DRIVE IS AVAILABLE
6673  032012  104107                          ERROR   107                 ;DRIVE SIEZED BY OTHER PORT
6674  032014  000137  033236                  JMP     37$
6675  032020  032765  020000  000016  65$:    BIT     #SPAR,P.CS1(R5)     ;TEST D TO C PARITY ERROR
6676  032026  001406                          BEQ     6$
6677  032030  104006                          ERROR   6
6678  032032  052737  004000  005424          BIS     #RCLREQ,RECODE
6679  032040  000137  033236                  JMP     37$
6680  032044  032704  000010          6$:     BIT     #DRPAR,R4           ;TEST DRIVE DETECTED PARITY ERROR
6681  032050  001406                          BEQ     7$
6682  032052  104007                          ERROR   7
6683  032054  052737  004000  005424          BIS     #RCLREQ,RECODE
6684  032062  000137  033236                  JMP     37$
6685  032066  032765  000010  000036  7$:     BIT     #ACLO,P.DS(R5)      ;TEST AC LOW
6686  032074  001403                          BEQ     8$
6687  032076  104010                          ERROR   10
6688  032100  000137  033236                  JMP     37$
```

```
 6689  032104  032765  000020  000036  8$:     BIT     #SPDLSS,P.DS(R5) ;TEST SPEED LOSS
 6690  032112  001403                          BEQ     24$
 6691  032114  104011                          ERROR   11
 6692  032116  000137  033236                  JMP     37$
 6693  032122  032765  004000  000016  24$:    BIT     #CTO,P.CS1(R5)  ;TEST FOR CONTROLLER TIMEOUT
 6694  032130  001401                          BEQ     25$
 6695  032132  104027                          ERROR   27
 6696  032134  032704  000001          25$:    BIT     #ILC,R4         ;TEST ILLEGAL FUNCTION CODE
 6697  032140  001403                          BEQ     10$
 6698  032142  104012                          ERROR   12
 6699  032144  000137  033236                  JMP     37$
 6700  032150  032765  002000  000020  10$:    BIT     #PGE,P.CS2(R5)  ;TEST PROGRAMMING ERROR
 6701  032156  001403                          BEQ     11$
 6702  032160  104013                          ERROR   13
 6703  032162  000137  033236                  JMP     37$
 6704  032166  032704  000004          11$:    BIT     #ILF,R4         ;TEST ILLEGAL DRIVE FUNCTION
 6705  032172  001403                          BEQ     12$
 6706  032174  104014                          ERROR   14
 6707  032176  000137  033236                  JMP     37$
 6708  032202  032704  000040          12$:    BIT     #DTYE,R4        ;TEST DRIVE TYPE ERROR
 6709  032206  001403                          BEQ     13$
 6710  032210  104015                          ERROR   15
 6711  032212  000137  033236                  JMP     37$
 6712  032216  032704  000020          13$:    BIT     #FMTE,R4        ;TEST FORMAT ERROR
 6713  032222  001403                          BEQ     14$
 6714  032224  104016                          ERROR   16
 6715  032226  000137  033236                  JMP     37$
 6716  032232  032704  004000          14$:    BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
 6717  032236  001403                          BEQ     15$
 6718  032240  104017                          ERROR   17
 6719  032242  000137  033236                  JMP     37$
 6720  032246  032704  040000          15$:    BIT     #UNS,R4         ;TEST DRIVE UNSAFE
 6721  032252  001406                          BEQ     16$
 6722  032254  104020                          ERROR   20
 6723  032256  052737  000200  005424          BIS     #ABORT,RECODE
 6724  032264  000137  033336                  JMP     ALLTRM
 6725  032270  032704  000002          16$:    BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
 6726  032274  001406                          BEQ     17$
 6727  032276  104021                          ERROR   21
 6728  032300  052737  004000  005424          BIS     #RCLREQ,RECODE
 6729  032306  000137  033236                  JMP     37$
 6730  032312  032704  001000          17$:    BIT     #COE,R4         ;TEST CYLINDER OVERFLOW
 6731  032316  001406                          BEQ     18$
 6732  032320  104022                          ERROR   22
 6733  032322  052737  004000  005424          BIS     #RCLREQ,RECODE
 6734  032330  000137  033236                  JMP     37$
 6735  032334  032704  002000          18$:    BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
 6736  032340  001406                          BEQ     19$
 6737  032342  104023                          ERROR   23
 6738  032344  052737  004000  005424          BIS     #RCLREQ,RECODE
 6739  032352  000137  033236                  JMP     37$
 6740  032356  032765  000040  000036  19$:    BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
 6741  032364  001406                          BEQ     20$
 6742  032366  104024                          ERROR   24
 6743  032370  052737  004000  005424          BIS     #RCLREQ,RECODE
 6744  032376  000137  033236                  JMP     37$
```

```
6745  032402  032704  010000          20$:    BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
6746  032406  001406                          BEQ     21$
6747  032410  104025                          ERROR   25
6748  032412  052737  000200  005424          BIS     #ABORT,RECODE
6749  032420  000137  033236                  JMP     37$
6750  032424  032765  100000  000020  21$:    BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
6751  032432  001403                          BEQ     22$
6752  032434  104026                          ERROR   26
6753  032436  000137  033236                  JMP     37$
6754  032442  032704  020000          22$:    BIT     #OPI,R4         ;TEST IF OPI ERROR
6755  032446  001470                          BEQ     29$
6756  032450  052737  000010  005424          BIS     #OPIERR,RECODE
6757  032456  105737  003130                  TSTB    DERCNT          ;TEST IF FIRST ERROR
6758  032462  001402                          BEQ     50$
6759  032464  000137  033236                  JMP     37$             ;NO - SKIP REPORT
6760  032470  032737  000400  005424  50$:    BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
6761  032476  001403                          BEQ     26$             ;HAS ALREADY OCCURRED
6762  032500  104054                  27$:    ERROR   54
6763  032502  000137  033236                  JMP     37$             ;GET OUT OF ERROR REPORT
6764  032506  004737  034444          26$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
6765  032512  004737  034134                  JSR     PC,RDHD0        ;GET HEADER OF SECTOR 0
6766  032516  032737  000400  005424          BIT     #LEV2ER,RECODE  ;TEST IF ERROR GETTING HDR
6767  032524  001036                          BNE     28$             ;IF YES-GO MAKE ABREVIATED REPORT
6768  032526  013702  003046                  MOV     RKBAS,R2        ;STORE HEADER 0 INTO REGISTERS
6769  032532  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;RESET INTERRUPT ENABLE
6770  032540  016237  000024  001202          MOV     RKDB(R2),$REG10 ;FOR REPORTING
6771  032546  016237  000024  001204          MOV     RKDB(R2),$REG11
6772  032554  016237  000024  001206          MOV     RKDB(R2),$REG12
6773  032562  032762  100000  000000          BIT     #CERR,RKCS1(R2) ;TEST IF ERROR DURING STORAGE
6774  032570  001343                          BNE     27$
6775  032572  105737  003143                  TSTB    TSTTYP          ;SEE IF READING WITH OFFSET
6776  032576  001403                          BEQ     54$             ;BR IF NOT
6777  032600  105037  003135                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
6778  032604  000401                          BR      55$             ;CONTINUE
6779  032606  104030                  54$:    ERROR   30              ;MAKE OPI REPORT
6780  032610  052737  004000  005424  55$:    BIS     #RCLREQ,RECODE  ;SET RECALIBRATE REQUEST BIT
6781  032616  000137  033236                  JMP     37$
6782  032622  104054                  28$:    ERROR   54
6783  032624  000137  031524                  JMP     ER2ENT          ;GO MAKE 2ND LEVEL REPORT
6784  032630  032704  000400          29$:    BIT     #HVRC,R4        ;TEST IF HVRC ERROR
6785  032634  001457                          BEQ     23$
6786  032636  052737  000004  005424          BIS     #HVRCER,RECODE
6787  032644  105737  003130                  TSTB    DERCNT          ;TEST IF FIRST ERROR
6788  032650  001402                          BEQ     30$             ;YES - REPORT
6789  032652  000137  033236                  JMP     37$             ;JUMP TO RETURN
6790  032656  032737  000400  005424  30$:    BIT     #LEV2ER,RECODE  ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
6791  032664  001403                          BEQ     31$             ;OCCURRED. NO-SKIP EXIT
6792  032666  104055                  51$:    ERROR   55
6793  032670  000137  033236                  JMP     37$             ;GET OUT OF ERROR REPORT
6794  032674  004737  034444          31$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
6795  032700  004737  034134                  JSR     PC,RDHD0        ;GO GET HDR 0
6796  032704  032737  000400  005424          BIT     #LEV2ER,RECODE  ;TEST IF ERROR IN GETTING HDR
6797  032712  001025                          BNE     32$             ;IF YES-GO MAKE ABREVIATED REPORT
6798  032714  013702  003046                  MOV     RKBAS,R2        ;GET RK611 BASE ADDRESS
6799  032720  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;CLEAR INTERRUPT ENABLE
6800  032726  016237  000024  001202          MOV     RKDB(R2),$REG10 ;STORE OFF HEADER
```

```
6801  032734  016237  000024  001204          MOV     RKDB(R2),$REG11
6802  032742  016237  000024  001206          MOV     RKDB(R2),$REG12
6803  032750  032762  100000  000000          BIT     #CERR,RKCS1(R2)  ;TEST IN ANY ERROR IN UNLOAD
6804  032756  001343                          BNE     51$              ;IY YES - GO MAKE SHORT REPORT
6805  032760  104031                          ERROR   31               ;MAKE FULL REPORT
6806  032762  000137  033236                  JMP     37$
6807  032766  104055                  32$:    ERROR   55               ;ABREVIATED HVRC ERROR RPORT
6808  032770  000137  031524                  JMP     ER2ENT           ;GO REPORT 2ND LEVEL ERROR
6809  032774  032704  000200          23$:    BIT     #BSE,R4          ;TEST FOR BAD SECTOR ERROR
6810  033000  001436                          BEQ     33$              ;NO - SKIP
6811  033002  052737  000002  005424          BIS     #BSERR,RECODE    ;SET ERROR FLAG
6812  033010  016500  000030                  MOV     P.DCYL(R5),R0    ;GET CYL NO.
6813  033014  116501  000027                  MOVB    P.DTS+1(R5),R1   ;GET TRACK NO.
6814  033020  042701  177774                  BIC     #177774,R1       ;CLEAR ALL BITS EXCEPT TRACK
6815  033024  016502  000026                  MOV     P.DTS(R5),R2     ;GET SECTOR IN ERROR
6816  033030  042702  177740                  BIC     #177740,R2       ;CLEAR ALL BITS EXCEPT SECTOR
6817  033034  004737  030042                  JSR     PC,BDSRCK        ;GO SEE IF THIS SECTOR LISTED
6818  033040  032737  001000  005424          BIT     #BADSEC,RECODE   ;TEST RESULT
6819  033046  001073                          BNE     37$              ;YES - EXIT, NO ERROR OR REPORT
6820  033050  105737  003143                  TSTB    TSTTYP           ;SEE IF READING WITH OFFSET
6821  033054  001403                          BEQ     56$              ;BR IF NOT
6822  033056  105037  003135                  CLRB    ERRCNT           ;CLEAR ERROR COUNT
6823  033062  000465                          BR      37$              ;CONTINUE
6824  033064  104104                  56$:    ERROR   104              ;ELSE REPORT BAD BSE
6825  033066  042737  000002  005424          BIC     #BSERR,RECODE    ;RESET BSE ERROR FLAG
6826  033074  000460                          BR      37$              ;GO EXIT
6827  033076  032704  100000          33$:    BIT     #DCK,R4          ;TEST IF DATA CHECK
6828  033102  001435                          BEQ     36$
6829  033104  052737  000020  005424          BIS     #DCKERR,RECODE   ;SET DATA CHECK ERROR IN RECOVERY CODE
6830  033112  032704  000100                  BIT     #ECH,R4          ;TEST IF ECC IS HARD. IF
6831  033116  001406                          BEQ     34$              ;YES SET UNCORRECTABLE IN
6832  033120  052737  000040  005424          BIS     #ECCNC,RECODE    ;RECOVERY FLAG AND A 0 IN
6833  033126  005037  001206                  CLR     $REG12           ;REG12 TO INDICATE UNCORRECTABLE,
6834  033132  000403                          BR      35$              ;A 1 IN REG 12 FOR CORRECTABLE
6835  033134  012737  000001  001206  34$:    MOV     #1,$REG12
6836  033142  105737  003130          35$:    TSTB    DERCNT           ;TEST IF FIRST ERROR
6837  033146  001033                          BNE     37$              ;NO SKIP REPORT
6838  033150  004737  034310                  JSR     PC,GTPKAD        ;GO GET PACK ADDRESS OF ERROR
6839  033154  016537  000060  001202          MOV     P.EPOS(R5),$REG10 ;STORE ECC POSITION &
6840  033162  016537  000062  001204          MOV     P.EPAT(R5),$REG11 ;PATTERN
6841  033170  104032                          ERROR   32               ;REPORT DCK ERROR
6842  033172  000137  033236                  JMP     37$
6843  033176  032765  040000  000020  36$:    BIT     #WCE,P.CS2(R5)   ;TEST WRITE CHECK ERROR
6844  033204  001414                          BEQ     37$
6845  033206  042737  000200  005424          BIC     #ABORT,RECODE    ;CLEAR ABORT & SET WRITE
6846  033214  052737  000100  005424          BIS     #WCERR,RECODE    ;CHECK ERROR IN RECODE
6847  033222  105737  003130                  TSTB    DERCNT           ;TEST IF FIRST ERROR
6848  033226  001003                          BNE     37$              ;NO - SKIP
6849  033230  004737  034310                  JSR     PC,GTPKAD        ;GO GET ADDRESS OF ERROR
6850  033234  104033                          ERROR   33               ;REPORT WCE
6851
6852  033236  032765  000020  000014  37$:    BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
6853  033244  001404                          BEQ     43$
6854  033246  104036                          ERROR   36
6855  033250  052737  000200  005424          BIS     #ABORT,RECODE
6856
```

```
6857  033256  032765  000040  000014  43$:    BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
6858  033264  001404                          BEQ     44$
6859  033266  104037                          ERROR   37
6860  033270  052737  000200  005424          BIS     #ABORT,RECODE
6861
6862  033276  032765  004000  000014  44$:    BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
6863  033304  001404                          BEQ     46$
6864  033306  104040                          ERROR   40
6865  033310  052737  000200  005424          BIS     #ABORT,RECODE
6866
6867  033316  032765  000010  000014  46$:    BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
6868  033324  001404                          BEQ     ALLTRM
6869  033326  104042                          ERROR   42
6870  033330  052737  000200  005424          BIS     #ABORT,RECODE
6871
6872
6873                                  ;ALL ERRORS MUST EXIT THROUGH THIS POINT
6874
6875  033336  012705  002640          ALLTRM: MOV     #PARM0,R5           ;RESTORE PARAMETER BLOCK SELECTION
6876  033342  012737  020556  003056          MOV     #DCKHDL,A.ABNL      ;SET READ ERROR HANDLER
6877  033350  105737  003143                  TSTB    TSTTYP              ;SEE IF READING WITH OFFSETS
6878  033354  001003                          BNE     20$                 ;BR IF YES
6879  033356  012737  031504  003056          MOV     #ERRHDL,A.ABNL      ;RESTORE INTERRUPT VECTORS FOR RETRY
6880  033364  012737  030272  003054  20$:    MOV     #ERRFRE,A.NORM      ;DRIVER OPERATIONS, IF ANY
6881  033372  032737  000200  005424          BIT     #ABORT,RECODE       ;IF ABORT IS NOT SET AND
6882  033400  001057                          BNE     48$                 ;THE DRIVE IS READY (HAS NOT
6883                                                                      ;CYCLED DOWN)
6884  033402  013702  003046                  MOV     RKBAS,R2            ;GET BASE ADDRESS
6885  033406  032762  000200  000012          BIT     #RDY,RKDS(R2)       ;TEST IF DRIVE READY SET
6886  033414  001004                          BNE     47$                 ;RECALIBRATE REQUIRED BIT IS SET
6887  033416  052737  000200  005424          BIS     #ABORT,RECODE       ;ELSE ABORT WITH ABORT MESSAGE
6888  033424  000445                          BR      48$
6889  033426  032737  004000  005424  47$:    BIT     #RCLREQ,RECODE      ;IF RECALIBRATE IS REQUIRED
6890  033434  001451                          BEQ     BGNRTY              ;FOR RETRY SET UP PARAM
6891  033436  013737  002640  002724          MOV     PARM0,PARM1         ;SAVE COMMAND
6892  033444  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;BLOCK TO DO IT.
6893  033452  012737  033736  003056          MOV     #RETANL,A.ABNL
6894  033460  004737  030146                  JSR     PC,DRVCAL
6895  033464  013737  002724  002640          MOV     PARM1,PARM0         ;RESTORE COMMAND
6896  033472  012737  020556  003056          MOV     #DCKHDL,A.ABNL      ;SET READ ERROR HANDLER
6897  033500  105737  003143                  TSTB    TSTTYP              ;SEE IF READING WITH OFFSETS
6898  033504  001003                          BNE     50$                 ;BR IF YES
6899  033506  012737  031504  003056          MOV     #ERRHDL,A.ABNL      ;RESTORE ERROR RETURN
6900  033514  032737  000400  005424  50$:    BIT     #LEV2ER,RECODE      ;IF AN ERROR OCCURRED IN THE
6901  033522  001416                          BEQ     BGNRTY              ;RECAL ATTEMP SET ABORT,
6902  033524  052737  000200  005424          BIS     #ABORT,RECODE       ;PRINT THE RECAL ERROR MESSAGE, AND
6903  033532  104060                          ERROR   60                  ;GO REPORT DETAILS
6904  033534  000137  031524                  JMP     ER2ENT
6905  033540  104061                  48$:    ERROR   61                  ;REPORT ABORT-RETRY FAILED
6906
6907                                  ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
6908                                  ;IF THE DRIVE READY BIT IS RESET.
6909
6910  033542  000005                  HLTPRG: RESET                       ;DISABLE ALL DEVICES
6911  033544  000000                          HALT                        ;HALT THE CPU
6912  033546  105037  003135                  CLRB    ERRCNT              ;CLEAR ERROR COUNT
```

```
                                        RESET                           ;RESET ALL DEVICES
                                        JMP     CMSTRT
                                ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
                                ;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.
033560  032737  000136  005424  BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
033566  001404                          BEQ     3$
033570  052737  100000  005424  9$:     BIS     #ANYDER,RECODE
033576  000453                          BR      2$
033600                          3$:
033600  105737  003150                  TSTB    NORTRY                  ;SEE IF "NO-RETRY" FLAG SET
033604  001371                          BNE     9$                      ;BR IF YES
033606  032737  001000  005424          BIT     #BADSEC,RECODE          ;TEST IF BAD SECTOR FLAG SET
033614  001044                          BNE     2$                      ;IF YES-EXIT TO CALLER
033616  123737  003135  003136          CMPB    ERRCNT,ERRLMT           ;BEGIN RETRY IF ERROR COUNT HAS
033624  001012                          BNE     1$                      ;NOT BEEN EXCEEDED
033626  005037  001174                  CLR     $REG5
033632  113737  003135  001174          MOVB    ERRCNT,$REG5            ;GET ERROR RETRY COUNT
033640  104102                          ERROR   102                     ;REPORT RETRY UNSUCCESSFUL
033642  052737  000200  005424          BIS     #ABORT,RECODE           ;SET ABORT & QUIT
033650  000734                          BR      HLTPRG
033652  013702  003046          1$:     MOV     RKBAS,R2                ;GET RK BASE ADDRESS
033656  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
033664  004737  030146                  JSR     PC,DRVCAL               ;GO DO IT
033670  012700  005176                  MOV     #COMSTR,R0              ;GO AND REESTABLISH THE COMMAND
033674  012025                          MOV     (R0)+,(R5)+             ;AS IT WAS ENTERED INTO THE
033676  012025                          MOV     (R0)+,(R5)+             ;PARAMETER BLOCK
033700  012025                          MOV     (R0)+,(R5)+
033702  012025                          MOV     (R0)+,(R5)+
033704  012025                          MOV     (R0)+,(R5)+
033706  012025                          MOV     (R0)+,(R5)+
033710  012705  002640                  MOV     #PARM0,R5
033714  012737  000000  177776          MOV     #PRO,@#PSW              ;LOWER PRIORITY TO ALLOW INTERRUPT
033722  004737  030146                  JSR     PC,DRVCAL               ;CALL DRIVER
033726  105037  003135          2$:     CLRB    ERRCNT                  ;CLEAR ERROR COUNT
033732  104410                          RESREG                          ;IF RETURN GETS HERE, NO ERROR
033734  000207                          RTS     PC                      ;OCCURRED, RECOVERY WAS SUCCESSFUL

033736  152737  000377  003132  RETANL: BISB    #377,DONE               ;SET DONE
033744  052737  000400  005424          BIS     #LEV2ER,RECODE          ;SET LEVEL TWO ERROR
033752  000207                          RTS     PC
033754  152737  000377  003132  RETNML: BISB    #377,DONE               ;SET DONE
033762  000207                          RTS     PC

                                ;;*****************************************************************
                                ;SBTTL  TIME OUT PROCESSOR ROUTINE
                                ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
                                ;*GATHERING DUTIES.
                                ;;*****************************************************************
```

```
 6969  033764                                TOPROC:
 6970  033764  104407                              SAVREG
 6971  033766  013702  003046                      MOV     RKBAS,R2
 6972  033772  012701  001174                      MOV     #$REGS,R1       ;SET UP R1 FOR RK REGISTER STORAGE
 6973  033776  016221  000000                      MOV     RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
 6974  034002  016221  000010                      MOV     RKCS2(R2),(R1)+ ;REGISTERS
 6975  034006  016221  000020                      MOV     RKDC(R2),(R1)+
 6976  034012  016221  000006                      MOV     RKDA(R2),(R1)+
 6977  034016  016221  000002                      MOV     RKWC(R2),(R1)+
 6978  034022  016221  000004                      MOV     RKBA(R2),(R1)+
 6979  034026  016221  000016                      MOV     RKASOF(R2),(R1)+
 6980  034032  016221  000012                      MOV     RKDS(R2),(R1)+
 6981  034036  016221  000014                      MOV     RKER(R2),(R1)+
 6982  034042  005000                              CLR     R0              ;THIS CODE WILL ATTEMPT TO
 6983                                                                      ;RETRIEVE THE STATUS FROM THE
 6984                                                                      ;DRIVE.
 6985  034044  012705  002724                      MOV     #PARM1,R5       ;SET UP TO USE PARM1
 6986  034050  112765  000141  000001              MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
 6987  034056  004737  030146                      JSR     PC,DRVCAL       ;CALL DRIVER
 6988  034062  032765  000100  000014              BIT     #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
 6989  034070  001403                              BEQ     1$              ;NO - SKIP
 6990  034072  052737  002000  005424              BIS     #TWOTOS,RECODE  ;SET TWO TIMEOUTS FLAG
 6991  034100  062705  000040              1$:     ADD     #P.ADD,R5       ;BUMP R5 TO POINT TO DRIVE STATUS
 6992  034104  012521                              MOV     (R5)+,(R1)+     ;MOVE ALL THE DRIVE STATUS INTO THE
 6993  034106  012521                              MOV     (R5)+,(R1)+     ;TEMP REGS FOR REPORTING.
 6994  034110  012521                              MOV     (R5)+,(R1)+
 6995  034112  012521                              MOV     (R5)+,(R1)+
 6996  034114  012521                              MOV     (R5)+,(R1)+
 6997  034116  012521                              MOV     (R5)+,(R1)+
 6998  034120  012521                              MOV     (R5)+,(R1)+
 6999  034122  012521                              MOV     (R5)+,(R1)+
 7000
 7001  034124  012705  002640                      MOV     #PARM0,R5       ;RESTORE PARM 0
 7002  034130  104410                              RESREG
 7003  034132  000207                              RTS     PC
 7004
 7005
 7006
 7007                                  ;;*****************************************************************
 7008                                  .SBTTL  READ HEADER 0 ROUTINE
 7009                                  ;;*****************************************************************
 7010  034134                                RDHD0:
 7011  034134  104407                              SAVREG
 7012  034136  016501  000026                      MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
 7013  034142  016500  000052                      MOV     P.B10(R5),R0    ;GET THE CYLINDER ADRS
 7014  034146  042700  160017                      BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
 7015  034152  006200                              ASR     R0              ;OFF UNUSED BITS AND POSITION
 7016  034154  006200                              ASR     R0              ;FOR USE AS THE DESIRED
 7017  034156  006200                              ASR     R0              ;CYLINDER IN THE READ
 7018  034160  006200                              ASR     R0              ;HEADER COMMAND.
 7019  034162  012705  002724                      MOV     #PARM1,R5       ;SET UP TO USE P.B. 1
 7020  034166  010165  000004                      MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
 7021  034172  010065  000002                      MOV     R0,P.CYLN(R5)   ;SET CYL NO.
 7022  034176  112765  000177  000001              MOVB    #SUBCLR,P.CMND(R5)      ;SET S.S. CLEAR CMND
 7023  034204  012737  000000  177776              MOV     #PRO,@#PSW      ;ALLOW INTERRUPTS
 7024  034212  004737  030146                      JSR     PC,DRVCAL       ;INIT THE S.S.
```

```
 7025  034216  112765  000125  000001          MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
 7026  034224  004737  030146                  JSR     PC,DRVCAL          ;DO A READ HEADER
 7027  034230  013762  024406  000000          MOV     HOLD1,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 0
 7028  034236  032762  000200  000000  24$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
 7029  034244  001774                          BEQ     24$
 7030  034246  013762  024410  000000          MOV     HOLD2,RKCS1(R2) ;DO A READ HDR, WITH FMT BIT = 1
 7031  034254  032762  000200  000000  26$:    BIT     #RDY,RKCS1(R2)  ;WAIT FOR READY
 7032  034262  001774                          BEQ     26$
 7033  034264  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
 7034  034272  153765  003134  000007          BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
 7035  034300  012705  002640                  MOV     #PARM0,R5          ;RESTORE P.B. 0 ADDRESS
 7036  034304  104410                          RESREG                     ;RESTORE R0-R5
 7037  034306  000207                          RTS     PC                 ;RETURN
 7038
 7039
 7040
 7041                                  ;;*****************************************************************
 7042                                  .SBTTL  GET PACK ADDRESS ROUTINE
 7043                                  ;;*****************************************************************
 7044  034310  016537  000030  001174  GTPKAD: MOV     P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
 7045  034316  005037  001176                  CLR     $REG6              ;CLEAR REGISTERS FOR
 7046  034322  005037  001200                  CLR     $REG7              ;TRACK & SECTOR STORAGE
 7047  034326  116537  000026  001200          MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
 7048  034334  116537  000027  001176          MOVB    P.DTS+1(R5),$REG6
 7049  034342  032737  000004  005424          BIT     #HVRCER,RECODE  ;SEE IF HVRC ERROR
 7050  034350  001034                          BNE     5$              ;BR IF YES
 7051  034352  005737  001200                  TST     $REG7             ;ADJUST THE ADDRESS CONTAINED IN
 7052                                                                  ;THE RK REGISTERS FOR THE AUTOMATIC
 7053  034356  001403                          BEQ     1$              ;INCREMENT
 7054  034360  005337  001200                  DEC     $REG7
 7055  034364  000426                          BR      5$
 7056  034366  032765  010000  000016  1$:     BIT     #CFMT,P.CS1(R5)
 7057  034374  001404                          BEQ     2$
 7058  034376  012737  000023  001200          MOV     #19.,$REG7
 7059  034404  000403                          BR      3$
 7060  034406  012737  000025  001200  2$:     MOV     #21.,$REG7
 7061  034414  005737  001176          3$:     TST     $REG6
 7062  034420  001403                          BEQ     4$
 7063  034422  005337  001176                  DEC     $REG6
 7064  034426  000405                          BR      5$
 7065  034430  012737  000002  001176  4$:     MOV     #2,$REG6
 7066  034436  005337  001174                  DEC     $REG5
 7067  034442  000207                  5$:     RTS     PC
 7068
 7069
 7070
 7071                                  ;;*****************************************************************
 7072                                  .SBTTL  BUILD EXPECTED HEADER
 7073                                  ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
 7074                                  ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
 7075                                  ;*7 FOR REPORTING.
 7076                                  ;;*****************************************************************
 7077  034444  104407                  BLDEXH: SAVREG
 7078  034446  016537  000030  001174          MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
 7079  034454  016501  000026                  MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
 7080  034460  042701  174377                  BIC     #174377,R1      ;CLEAR ALL BUT TRACK BITS
```

```
7081  034464  006201                        ASR     R1              ;AND SECTOR. SHIFT THE TRACK
7082  034466  006201                        ASR     R1              ;OVER TO CONFORM TO HEADER FORMAT
7083  034470  006201                        ASR     R1              ;CHECK THE FORMAT BIT AND
7084                                                                ;IF SET, SET THE HEADER FORMAT
7085  034472  016537  000026  001176        MOV     P.DTS(R5),$REG6 ;BIT.
7086  034500  042737  177740  001176        BIC     #177740,$REG6   ;CLEAR ALL BUT SECTOR
7087  034506  060137  001176                ADD     R1,$REG6        ;ADD TRACK AND SECTOR TOGETHER
7088  034512  052737  140000  001176        BIS     #140000,$REG6   ;INSERT BSE BITS
7089  034520  032765  010000  000016        BIT     #CFMT,P.CS1(R5)
7090  034526  001403                        BEQ     23$
7091  034530  052737  001000  001176        BIS     #1000,$REG6
7092  034536  013737  001174  001200  23$:  MOV     $REG5,$REG7     ;COMPUTE THE HEADER VRC
7093  034544  013701  001176                MOV     $REG6,R1
7094  034550  043737  001176  001200        BIC     $REG6,$REG7
7095  034556  043701  001174                BIC     $REG5,R1
7096  034562  050137  001200                BIS     R1,$REG7
7097  034566  104410                        RESREG
7098  034570  000207                        RTS     PC
7099
```

```
                                .SBTTL  RK611/RK06-RK07 UNIBUS DRIVER FOR SEQUENTIAL OPERATIONS (REV. 0.10)

                                ;*COPYRIGHT (C) 1975,1976,1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MA. 01754
                                ;*AUTHOR: ROY SPITZER

                                .SBTTL   *WATCH-DOG TIMER

                                ;;*****************************************************************
                                ;*
                                ;*      THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06-RK07 UNIBUS
                                ;*      SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
                                ;*      REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
                                ;*      THE RK06-RK07 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.
                                ;*
                                ;;*****************************************************************

034572  010546          W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
034574  010446                  MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
034576  010346                  MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
034600  010246                  MOV     R2,-(SP)        ;SAVE R2 ON STACK
034602  013746  177776          MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
034606  005337  003066          DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
034612  001034                  BNE     20$             ;IF NOT ZERO RETURN
034614  013737  003070  003066  MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
034622  105737  003110          TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
034626  001426                  BEQ     20$             ;NO, RETURN
034630  013737  003052  177776  MOV     RKPRI,PS        ;LOCK OUT RK06-RK07 INTERRUPTS
034636  013702  003046          MOV     RKBAS,R2        ;LOAD BASE OF RK06-RK07 REGISTERS
034642  005337  003124          DEC     W.DRV           ;DECREMENT COMMAND TIMER
034646  001016                  BNE     20$             ;RETURN IF NO TIME OUT
```

```
7156  034650  105037  003110                CLRB    W.TIME          ;RESET TIMING INDICATOR
7157  034654  013705  003122                MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
7158                                                                  ;  TABLE FOR INDEXING
7159  034660  052765  000100  000014        BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
7160  034666  020537  003064                CMP     R5,0.WAIT       ;CHECK IF DRIVER IS WAITING FOR
7161                                                                  ;  COMMAND COMPLETION
7162  034672  001002                        BNE     5$              ;NO, DO NOT ALTER WAITING FOR
7163                                                                  ;  COMMAND COMPLETION
7164  034674  005037  003064                CLR     0.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
7165  034700  004737  040154        5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
7166  034704  012637  177776        20$:    MOV     (SP)+,PS        ;RESTORE PSW
7167  034710  012602                        MOV     (SP)+,R2        ;RESTORE R2
7168  034712  012603                        MOV     (SP)+,R3        ;RESTORE R3
7169  034714  012604                        MOV     (SP)+,R4        ;RESTORE R4
7170  034716  012605                        MOV     (SP)+,R5        ;RESTORE R5
7171  034720  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
                        ;*
                        ;;*****************************************************************

034722  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
034724  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
034726  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
034730  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
034732  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
034734  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
034736  013702  003046          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
034742  016237  000010  003012  MOV     RKCS2(R2),T.CS2 ;STORE CS2
034750  032737  001000  003012  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
034756  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
034760  052737  100000  003062  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
034766  004737  040200          JSR     PC,R.CONT       ;REPORT ERROR
```

```
7228  034772  000137  037152                    JMP     I.RTRN          ;RETURN
7229
7230  034776  105737  003104          1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
7231  035002  001410                             BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
7232  035004  100403                             BMI     5$              ;CHECK IF RELEASE COMMAND
7233  035006  105037  003104                    CLRB    I.ISRL          ;YES,CLEAR FLAG
7234  035012  000473                             BR      I.IGO           ;CONTINUE PROCESSING INTERRUPT
7235
7236  035014  105037  003104          5$:       CLRB    I.ISRL          ;CLEAR FLAG
7237  035020  000137  036134                    JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
7238
7239  035024  032737  010400  003012  6$:       BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
7240                                                                    ;  UNIT FIELD ERROR
7241  035032  001413                             BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
7242  035034  013704  003012                    MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
7243  035040  042704  177770                    BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
7244  035044  013705  003122                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
7245  035050  016237  000000  003010            MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
7246  035056  000137  035344                    JMP     I.ERRC          ;REPORT ERROR
7247
7248  035062  016237  000012  003030  7$:       MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
7249  035070  032737  000001  003030            BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
7250                                                                    ;  PORT
7251  035076  001041                             BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
7252
7253                                            ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
7254  035100  032737  164000  003012            BIT     #DLT!WCE!UPE!NEM,T.CS2
7255
7256  035106  001007                             BNE     10$             ;INDICATE ERROR
7257  035110  016237  000014  003026            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
7258
7259                                  ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
7260  035116  032737  125700  003026            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
7261
7262  035124  001407                             BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
7263
7264  035126  052737  000010  003062  10$:       BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
7265  035134  004737  040200                    JSR     PC,R.CONT       ;REPORT ERROR
7266  035140  000137  037152                    JMP     I.RTRN          ;RESTORE REGISTERS
7267
7268  035144  105037  003110          11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
7269  035150  005037  003124                    CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
7270  035154  013705  003122                    MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
7271                                                                    ;  ADDRESS
7272  035160  052765  010000  000014            BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
7273                                                                    ;  PROGRAM DRIVE STATUS REGISTER
7274  035166  005037  003064                    CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
7275  035172  004737  040154                    JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
7276  035176  000137  037152                    JMP     I.RTRN          ;GO RESTORE REGISTERS
7277
7278  035202  013705  003064          I.IOO:    MOV     O.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
7279  035206  001002                             BNE     2$              ;IS COMMAND WAITING PROCESSING
7280                                                                    ;  YES, DO PROCESSING
7281  035210  000137  036134                    JMP     I.ATTN          ;NO, PROCESS ATTENTION
7282
7283  035214  013704  003012          2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
7284  035220  042704  177770                    BIC     #↑C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
7285
7286
7287  035224  126504  000000                    CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
7288  035230  001401                            BEQ     3$              ;YES, CONTINUE
7289  035232  000000                            HALT                    ;NO, DRIVER ERROR
7290  035234  122765  000164  000001  3$:       CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
7291  035242  001002                            BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
7292  035244  000137  035602                    JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
7293
7294  035250  005037  003064          10$:      CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
7295  035254  005037  003124                    CLR     W.DRV           ;CLEAR WATCH-DOG TIME
7296  035260  105037  003110                    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
7297  035264  016237  000000  003010            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7298  035272  032737  100000  003010            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
7299  035300  001021                            BNE     I.ERRC          ;YES, PROCESS ERROR
7300  035302  016237  000016  003024            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7301  035310  133737  003111  003025            BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
7302  035316  001004                            BNE     15$             ;YES, REPORT ERROR
7303  035320  004737  040166                    JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
7304  035324  000137  037152                    JMP     I.RTRN          ;RESTORE REGISTERS
7305
7306  035330  052765  000010  000014  15$:      BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
7307
7308  035336  004737  037622          I.ERRA:   JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
7309  035342  000405                            BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
7310
7311  035344  013765  003010  000016  I.ERRC:   MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
7312  035352  004737  037644                    JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
7313  035356  016265  000032  000062  I.ERR:    MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
7314  035364  016265  000030  000060            MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
7315  035372  004037  037170                    JSR     R0,I.CCLR       ;CLEAR CONTROLLER
7316  035376  037152                            I.RTRN                  ;ERROR RETURN
7317  035400  032765  010400  000020            BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
7318                                                                    ;  UNIT FIELD ERROR
7319  035406  001046                            BNE     5$              ;YES, REPORT ERROR
7320  035410  004037  037726                    JSR     R0,I.STAT       ;GATHER DRIVE STATUS
7321  035414  037152                            I.RTRN                  ;ERROR RETURN
7322  035416  112737  000005  003010            MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
7323  035424  004037  037252                    JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
7324  035430  037152                            I.RTRN                  ;ERROR RETURN
7325  035432  133737  003111  003025            BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
7326  035440  001407                            BEQ     2$              ;NO, INDICATE DRIVE ERROR
7327  035442  052737  000020  003062            BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
7328                                                                    ;  WITH CLEAR
7329  035450  004737  040200                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
7330  035454  000137  037152                    JMP     I.RTRN          ;GO RESTORE REGISTERS
7331
7332  035460  032737  040000  003034  2$:       BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
7333  035466  001403                            BEQ     3$              ;YES, CHECK FAULT
7334  035470  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
7335  035476  032737  001000  003036  3$:       BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
7336  035504  001407                            BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
7337  035506  052737  002000  003062            BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
7338  035514  004737  040200                    JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
7339  035520  000137  037152                    JMP     I.RTRN          ;RETURN
```

```
7340
7341  035524  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
7342  035532  001017                          BNE     10$             ;YES, GO REPORT ERROR
7343  035534  032737  020000  003034          BIT     #S.PIP,T.MR2    ;CHECK IF DRIVE IS CYCLING DOWN
7344  035542  001413                          BEQ     10$             ;NO, REPORT ERROR
7345  035544  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
7346  035552  113737  003111  003110          MOVB    INTMSK,W.TIME   ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
7347  035560  013737  003074  003124          MOV     W.8SEC,W.DRV
7348  035566  000137  037152                  JMP     I.RTRN          ;GO RESTORE REGISTERS
7349
7350  035572  004737  040154          10$:    JSR     PC,R.ABNL       ;GO REPORT ERROR
7351  035576  000137  037152                  JMP     I.RTRN          ;GO RESTORE REGISTERS
7352
7353                                  .SBTTL   *READ ALL HEADERS INTERRUPT SEQUENCE
7354
7355  035602  016237  000000  003010  I.HDAL: MOV     RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
7356                                                                  ;  ERROR
7357  035610  032737  100000  003010          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
7358  035616  001422                          BEQ     5$              ;NO, CHECK FOR ATTENTION
7359
7360  035620  005037  003064                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETE
7361  035624  105037  003110                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
7362  035630  005037  003124                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
7363  035634  013765  003010  000016          MOV     T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
7364  035642  004737  037644                  JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
7365  035646  004037  037170                  JSR     R0,I.CCLR       ;CLEAR CONTROLLER
7366  035652  037152                          I.RTRN                  ;ERROR RETURN
7367  035654  004737  040154                  JSR     PC,R.ABNL       ;INDICATE ERROR RETURN
7368  035660  000137  037152                  JMP     I.RTRN          ;RESTORE REGISTERS
7369
7370  035664  016237  000016  003024  5$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7371  035672  133737  003111  003025          BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
7372  035700  001410                          BEQ     7$              ;NO, CHECK IF READ ALL HEADERS
7373  035702  005037  003064                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETION
7374  035706  105037  003110                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
7375  035712  005037  003124                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
7376  035716  000137  035336                  JMP     I.ERRA          ;GO REPORT ERROR
7377
7378  035722  013701  003100          7$:     MOV     HDR.AD,R1       ;GET MAIN MEMORY ADDRESS
7379  035726  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET FIRST WORD OF HEADER
7380  035732  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET SECOND WORD OF HEADER
7381  035736  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET THIRD WORD OF HEADER
7382  035742  010137  003100                  MOV     R1,HDR.AD       ;STORE ADDRESS FOR NEXT HEADER
7383  035746  016237  000010  003012          MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
7384  035754  032737  100000  003012          BIT     #DLT,T.CS2      ;CHECK FOR DATA LATE
7385  035762  001055                          BNE     35$             ;YES, REPORT ERROR
7386  035764  005337  003102                  DEC     HDR.CT          ;DECREMENT NUMBER OF HEADER YET TO READ
7387  035770  001026                          BNE     25$             ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
7388  035772  005037  003064                  CLR     O.WAIT          ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
7389  035776  005037  003124                  CLR     W.DRV           ;CLEAR TIME OUT COUNT FOR THIS DRIVE
7390  036002  105037  003110                  CLRB    W.TIME          ;CLEAR WATCH DOG TIME ON THIS DRIVE
7391  036006  012762  000003  000026          MOV     #3,RKMR1(R2)    ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
7392  036014  112737  000001  003010          MOVB    #DR.SEL,T.CS1   ;LOAD SELECT COMMAND
7393  036022  004037  037252                  JSR     R0,I.ISSU       ;GET SECTOR COUNT
7394  036026  037152                          I.RTRN                  ;ERROR RETURN
7395  036030  013765  003036  000056          MOV     T.MR3,P.B11(R5) ;LOAD SECTOR COUNT
```

```
 7396  036036  004737  040166                      JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
 7397  036042  000137  037152                      JMP     I.RTRN          ;RESTORE REGISTERS
 7398
 7399  036046  016562  000002  000020  25$:        MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
 7400  036054  016562  000004  000006              MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
 7401  036062  116565  000007  000017              MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 7402  036070  042765  165777  000016              BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
 7403                                                      ;  DRIVE TYPE
 7404  036076  112765  000125  000016              MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
 7405  036104  016562  000016  000000              MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
 7406  036112  000137  037152                      JMP     I.RTRN          ;RESTORE REGISTERS
 7407
 7408  036116  052737  000400  003062  35$:        BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
 7409  036124  004737  040200                      JSR     PC,R.CONT       ;REPORT ERROR
 7410  036130  000137  037152                      JMP     I.RTRN          ;RESTORE REGISTERS
 7411
 7412                                      .SBTTL   *DRIVE ATTENTION SCANNER
 7413
 7414  036134  016237  000000  003010  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
 7415                                                      ;  REGISTER 1 FOR COMPARISON
 7416  036142  032737  100000  003010              BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
 7417  036150  001441                              BEQ     5$              ;NO, CHECK IF ATTENTION
 7418
 7419                                              ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
 7420  036152  032737  164000  003012              BIT     #DLT!WCE!UPE!NEM,T.CS2
 7421
 7422  036160  001007                              BNE     1$              ;INDICATE ERROR
 7423  036162  016237  000014  003026              MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
 7424
 7425                                      ;       CHECK FOR DATA TRANSFER ERROR TYPE
 7426  036170  032737  125700  003026              BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
 7427
 7428  036176  001407                              BEQ     2$              ;NO DATA TRANSFER ERROR
 7429
 7430  036200  052737  000010  003062  1$:         BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
 7431  036206  004737  040200                      JSR     PC,R.CONT       ;REPORT ERROR
 7432  036212  000137  037152                      JMP     I.RTRN          ;RESTORE REGISTERS
 7433
 7434  036216  013704  003012          2$:         MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
 7435  036222  042704  177770                      BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
 7436  036226  105037  003110                      CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
 7437  036232  005037  003124                      CLR     W.DRV           ;RESET TIMER VALUE
 7438  036236  013705  003122                      MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
 7439
 7440                                      ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
 7441                                      ;         IN PROGRAM DEVICE STATUS REGISTER
 7442  036242  042765  000006  000014              BIC     #DRVPOS!DRVPDT,P.PRST(R5)
 7443
 7444  036250  000137  035344                      JMP     I.ERRC          ;GO REPORT ERROR
 7445
 7446  036254  032737  040000  003010  5$:         BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
 7447  036262  001002                              BNE     6$              ;YES, PROCESS INTERRUPT
 7448  036264  000137  037152                      JMP     I.RTRN          ;RESTORE REGISTERS
 7449
 7450  036270  016237  000016  003024  6$:         MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 7451  036276  105737  003025                      TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
7452  036302  001007                        BNE     7$                      ;YES GO PROCESS INTERRUPT
7453  036304  052737  000002  003062        BIS     #E.NOAT,E.CONT          ;SET NO ATTENTION IN ATTENTION SUMMARY
7454  036312  004737  040200                JSR     PC,R.CONT               ;GO REPORT ERROR
7455  036316  000137  037152                JMP     I.RTRN                  ;GO RESTORE REGISTERS
7456
7457  036322  133737  003111  003025  7$:   BITB    INTMSK,T.ASOF+1         ;CHECK IF DESIRED INTERRUPT
7458  036330  001007                        BNE     8$                      ;YES, GO PROCESS IT
7459  036332  052737  000004  003062        BIS     #E.UATT,E.CONT          ;SET UNSOLICATED ATTENTION
7460  036340  004737  040200                JSR     PC,R.CONT               ;GO REPORT ERROR
7461  036344  000137  037152                JMP     I.RTRN                  ;GO RESTORE REGISTERS
7462
7463  036350  013705  003122          8$:   MOV     PBLKT,R5                ;STORE PARAMETER BLOCK TABLE
7464  036354  116504  000000                MOVB    P.DRVN(R5),R4           ;STORE DRIVE NUMBER
7465  036360  032765  020000  000014        BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
7466  036366  001402                        BEQ     11$                     ;NO, CONTINUE
7467  036370  000137  037052                JMP     I.UNLD                  ;SERVICE DRIVE IN POSITION AFTER ERROR
7468
7469  036374  042765  000002  000014  11$:   BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
7470  036402  005062  000026                CLR     RKMR1(R2)               ;CLEAR MAINTENANCE REGISTER 1
7471  036406  112737  000001  003010        MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
7472  036414  004037  037252                JSR     R0,I.ISSU               ;SELECT DRIVE WITH ATTENTION HIGH
7473  036420  037152                        I.RTRN                          ;ERROR RETURN
7474  036422  013765  003036  000042        MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
7475  036430  032765  000200  000042        BIT     #S.FLT,P.B00(R5)    ;CHECK IF DRIVE FAULT
7476  036436  001401                        BEQ     12$                     ;NO, CHECK FOR DRIVE STATUS CHANGE
7477  036440  000461                        BR      I.AERR                  ;PROCESS ERROR
7478
7479  036442  013765  003034  000040  12$:   MOV     T.MR2,P.A00(R5)         ;STORE MAINTENANCE REGISTER 2
7480  036450  032765  040000  000040        BIT     #S.DSC,P.A00(R5)    ;CHECK FOR DRIVE STATUS CHANGE
7481  036456  001004                        BNE     13$                     ;YES, PROCESS DRIVE STATUS CHANGE
7482  036460  052765  004000  000014        BIS     #NODSC,P.PRST(R5)   ;SET NO DRIVE STATUS CHANGE
7483  036466  000446                        BR      I.AERR                  ;PROCESS ERROR
7484
7485  036470  112737  000005  003010  13$:   MOVB    #DR.CLR,T.CS1           ;LOAD COMMAND
7486  036476  004037  037252                JSR     R0,I.ISSU               ;CLEAR DRIVE STATUS CHANGE
7487  036502  037152                        I.RTRN                          ;ERROR RETUN
7488  036504  013765  003024  000032        MOV     T.ASOF,P.ASOF(R5)   ;STORE ATTENTION SUMMARY
7489  036512  133765  003111  000033        BITB    INTMSK,P.ASOF+1(R5)  ;CHECK IF ATTENTION RESET
7490  036520  001407                        BEQ     15$                     ;YES, CONTINUE INTERRUPT PROCESSING
7491  036522  052737  000020  003062        BIS     #E.CLAT,E.CONT          ;SET ATTENTION DID NOT RESET
7492                                                                        ;  WITH DRIVE CLEAR
7493  036530  004737  040200                JSR     PC,R.CONT               ;FLAG ERROR
7494  036534  000137  037152                JMP     I.RTRN                  ;RESTORE REGISTERS
7495
7496  036540  013765  003034  000040  15$:   MOV     T.MR2,P.A00(R5)         ;STORE MAINTENANCE REGISTER 2
7497  036546  032765  040000  000040        BIT     #S.DSC,P.A00(R5)    ;CHECK IF DRIVE STATUS CHANGE
7498                                                                        ;  RESET
7499  036554  001404                        BEQ     16$                     ;YES, CONTINUE INTERRUPT PROCESSING
7500  036556  052765  000040  000014        BIS     #DRVDSC,P.PRST(R5)  ;SET DRIVE STATUS CHANGE DID NOT CLEAR
7501  036564  000407                        BR      I.AERR                  ;GO PROCESS ERROR
7502
7503  036566  105037  003110          16$:   CLRB    W.TIME                  ;RESET TIMING ON THIS DRIVE
7504  036572  005037  003124                CLR     W.DRV                   ;CLEAR DRIVE TIMING COUNT
7505  036576  004737  040166                JSR     PC,R.NORM               ;REPORT SUCCESSFUL COMMAND COMPLETETION
7506  036602  000563                        BR      I.RTRN                  ;RESTORE REGISTERS
7507
```

```
                                .SBTTL   *ATTENTION ERROR HANDLER
                        I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
                                                           ;  OF DATA TRANSFER
                                CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
                                CLR     W.DRV              ;RESET WATCH-DOG TIME
                                BIC     #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
                                BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
                                BIS     T.CS1,P.CS1(R5)    ;MAKE GOOD MESSAGE
                                MOV     T.CS2,P.CS2(R5)    ;STORE CONTROLLER REGISTERS
                                MOV     T.WCR,P.WCR(R5)
                                MOV     T.BA,P.BAR(R5)
                                MOV     T.DA,P.DTS(R5)
                                MOV     T.DC,P.DCYL(R5)
                                MOV     T.ASOF,P.ASOF(R5)
                                MOV     T.ER,P.ER(R5)
                                MOV     T.DS,P.DS(R5)
                                JSR     R0,I.STAT          ;GATHER DRIVE STATUS
                                I.RTRN                     ;ERROR RETURN
                                MOVB    #DR.CLR,T.CS1      ;LOAD COMMAND
                                JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
                                I.RTRN                     ;ERROR RETURN
                                BITB    INTMSK,T.ASOF+1    ;CHECK IF ATTENTION RESET
                                BEQ     2$                 ;YES, FLAG DRIVE ERROR
                                BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
                                JSR     PC,R.CONT          ;REPORT ERROR
                                JMP     I.RTRN             ;RESTORE REGISTERS

                        2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
                                BNE     10$                ;YES, REPORT ERROR
                                BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
                                BEQ     10$                ;NO, REPORT ERROR
                                BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
                                MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
                                MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
                                JMP     I.RTRN             ;RESTORE REGISTERS

                        10$:    JSR     PC,R.ABNL          ;REPORT ERROR
                                JMP     I.RTRN             ;RESTORE REGISTERS

                                .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD

                        I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
                                MOVB    #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
                                JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
                                I.RTRN                     ;ERROR RETURN
                                BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
                                BEQ     15$                ;YES, CONTINUE
                                MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
                                JSR     PC,R.CONT          ;REPORT ERROR
                                BR      I.RTRN             ;RESTORE REGISTERS

                        15$:    BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
                                BEQ     20$                ;YES, CONTINUE
                                BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
                        20$:    CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

```
 7564  037142  005037  003124                    CLR     W.DRV           ;CLEAR TIME COUNT
 7565  037146  004737  040154                    JSR     PC,R.ABNL       ;REPORT ERROR
 7566
 7567  037152  012600                  I.RTRN:   MOV     (SP)+,R0        ;RESTORE R0
 7568  037154  012601                            MOV     (SP)+,R1        ;RESTORE R1
 7569  037156  012602                            MOV     (SP)+,R2        ;RESTORE R2
 7570  037160  012603                            MOV     (SP)+,R3        ;RESTORE R3
 7571  037162  012604                            MOV     (SP)+,R4        ;RESTORE R4
 7572  037164  012605                            MOV     (SP)+,R5        ;RESTORE R5
 7573  037166  000002                            RTI                     ;RETURN
 7574
```

```
                                .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
                                ;;*
                                ;;*****************************************************************
037170  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
037176  016237  000000  003010          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
037204  032737  100000  003010          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
037212  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
037214  052737  000001  003062          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
037222  004737  040200                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
037226  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
037230  000200                          RTS     R0              ;RETURN

037232  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
037240  112737  177777  003104          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
037246  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
037250  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
                                ;*
                                ;;*****************************************************************

037252  013746  003010          I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
037256  005037  003012                  CLR     T.CS2           ;CLEAR TEMPORARY CS2
037262  116537  000000  003012        MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
037270  013762  003012  000010        MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
037276  116537  000007  003011        MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
037304  142737  177753  003011        BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                                                ;  FORMAT AND DRIVE TYPE
037312  013762  003010  000000        MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
037320  105762  000000          1$:   TSTB    RKCS1(R2)       ;WAIT FOR READY
037324  100375                        BPL     1$
037326  004737  037474                JSR     PC,I.STOR       ;GO STORE REGISTERS
037332  032737  100000  003010        BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
037340  001437                        BEQ     5$              ;NO, RETURN
037342  032737  001000  003012        BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
037350  001406                        BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
037352  052737  100000  003062        BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
037360  004737  040200                JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
037364  000440                        BR      10$             ;RETURN

                                ;CHECK IF ANY CONTROLLER ERROR IS SET
037366  032737  024000  003010  2$:   BIT     #CTO!SPAR,T.CS1
037374  001027                        BNE     7$
037376  032737  176400  003012        BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
037404  001023                        BNE     7$
037406  032737  131761  003026        BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
037414  001017                        BNE     7$

037416  122716  000005                CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
 7666  037422  001003                          BNE     3$                  ;NO, DO NOT SET DRIVE HARD ERROR
 7667  037424  052765  000020  000014          BIS     #DRVHRD,P.PRST(R5)  ;SET HARD DRIVE ERROR
 7668  037432  004037  037170          3$:     JSR     R0,I.CCLR           ;GO ISSUE A CONTROLLER CLEAR
 7669  037436  037466                          10$                         ;ERROR RETURN
 7670  037440  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)       ;SET INTERRUPT ENABLE
 7671  037446  005726                          TST     (SP)+               ;ADJUST STACK
 7672  037450  005720                          TST     (R0)+               ;ADJUST R0 FOR NORMAL RETURN
 7673  037452  000200                          RTS     R0                  ;RETURN
 7674
 7675  037454  052737  001000  003062  7$:     BIS     #E.CERR,E.CONT      ;SET CONTROLLER ERROR DURING
 7676                                                                      ;  DRIVER SERVICING
 7677  037462  004737  040200                  JSR     PC,R.CONT           ;REPORT ERROR
 7678  037466  005726                  10$:    TST     (SP)+               ;ADJUST STACK
 7679  037470  011000                          MOV     (R0),R0             ;ADJUST R0 FOR ERROR RETURN
 7680  037472  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
                                        ;*
                                        ;;**********************************************************************

 037474  016237  000000  003010  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
 037502  016237  000010  003012          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
 037510  016237  000002  003014          MOV     RKWC(R2),T.WCR
 037516  016237  000004  003016          MOV     RKBA(R2),T.BA
 037524  016237  000006  003020          MOV     RKDA(R2),T.DA
 037532  016237  000012  003030          MOV     RKDS(R2),T.DS
 037540  016237  000014  003026          MOV     RKER(R2),T.ER
 037546  016237  000016  003024          MOV     RKASOF(R2),T.ASOF
 037554  016237  000020  003022          MOV     RKDCYL(R2),T.DC
 037562  016237  000026  003032          MOV     RKMR1(R2),T.MR1
 037570  016237  000034  003034          MOV     RKMR2(R2),T.MR2
 037576  016237  000036  003036          MOV     RKMR3(R2),T.MR3
 037604  016237  000030  003040          MOV     RKECPS(R2),T.POS
 037612  016237  000032  003042          MOV     RKECPT(R2),T.PAT
 037620  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
                                ;*
                                ;;*****************************************************************

037622  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
                                                                        ;OF LAST COMMAND ISSUED
037630  042737  000036  003010          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
037636  053765  003010  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
037644  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
037652  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
037660  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
037666  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
037674  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
037702  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
037710  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
                                                                        ;  OFFSET
037716  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
037724  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
                                ;*
                                ;;*****************************************************************

037726  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 01
037734  112737  000001  003010          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
037742  004037  037252                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
037746  040136                          3$                              ;ERROR RETURN
037750  013765  003034  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
037756  013765  003036  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
037764  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 10
037772  112737  000001  003010          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
040000  004037  037252                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
040004  040136                          3$                              ;ERROR RETURN
040006  013765  003034  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
040014  013765  003036  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
040022  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
                                                                        ;  FOR STATUS BYTE 11
040030  112737  000001  003010          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
040036  004037  037252                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
040042  040136                          3$                              ;ERROR RETURN
040044  013765  003034  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
040052  013765  003036  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
040060  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 00
040064  112737  000001  003010          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
040072  004037  037252                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
040076  040136                          3$                              ;ERROR RETURN
040100  013765  003034  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
040106  013765  003036  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
040114  032737  001000  003036          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
040122  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
 7814  040124  052737  002000  003062          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
 7815  040132  004737  040200                  JSR     PC,R.CONT               ;REPORT ERROR
 7816  040136  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
 7817  040140  000200                          RTS     R0                      ;RETURN
 7818
 7819  040142  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
 7820  040150  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
 7821  040152  000200                          RTS     R0                      ;RETURN
 7822
```

```
                                        .SBTTL  *COMMON DRIVER RETURNS
                                
                                R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
                                        JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
                                        RTS     PC              ;RETURN
                                
                                R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
                                        JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
                                        RTS     PC              ;RETURN
                                
                                R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
                                        CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
                                        CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
                                        JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
                                        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
                                ;*
                                ;;*****************************************************************

040222  010546          C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
040224  010446                  MOV     R4,-(SP)        ;STORE R4 ON STACK
040226  010346                  MOV     R3,-(SP)        ;STORE R3 ON STACK
040230  010246                  MOV     R2,-(SP)        ;STORE R2 ON STACK
040232  010146                  MOV     R1,-(SP)        ;STORE R1 ON STACK
040234  010046                  MOV     R0,-(SP)        ;STORE R0 ON STACK
040236  013746  177776          MOV     PS,-(SP)        ;STORE PSW ON STACK
040242  013737  003052  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
040250  017605  000016          MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
040254  062766  000002  000016  ADD     #2,16(SP)       ;ADJUST RETURN
040262  016504  000000          MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
040266  042704  177770          BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
040272  010537  003122          MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
040276  116437  003112  003111  MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
040304  116437  003112  003110  MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
040312  013737  003072  003124  MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

040320  013702  003046          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
```

```
7894  040324  042765  075176  000014          BIC     #↑C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
7895
7896  040332  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
7897  040334  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
7898  040340  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
7899  040342  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
7900
7901  040346  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
7902  040350  020001                          CMP     R0,R1           ;CHECK IF FINISHED
7903  040352  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
7904  040354  105037  003104                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
7905  040360  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
7906  040364  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
7907  040370  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
7908  040376  001402                          BEQ     3$              ;NO, PROCESS
7909  040400  000137  041114                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
7910
7911  040404  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
7912                                                                  ;  START SPINDLE
7913                                                                  ;  RECALIBRATE
7914                                                                  ;  OFFSET
7915                                                                  ;  SEEK
7916                                                                  ;  UNLOAD
7917
7918  040412  101174                          BHI     25$             ;NO, DRIVE COMMAND
7919                                                                  ;  SELECT DRIVE
7920                                                                  ;  PACK ACKNOWLEDGE
7921                                                                  ;  CLEAR
7922
7923  040414  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
7924  040422  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
7925                                                                  ;  READ DATA
7926                                                                  ;  WRITE DATA
7927                                                                  ;  READ HEADER
7928                                                                  ;  WRITE HEADER
7929                                                                  ;  WRITE CHECK
7930  040424  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
7931  040432  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
7932  040440  005037  003064                  CLR     @.WAIT          ;CLEAR WAIT FOR COMMAND
7933  040444  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
7934  040452  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
7935  040454  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
7936  040462  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
7937  040470  000431                          BR      8$              ;GO ISSUE COMMAND
7938
7939  040472  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
7940  040500  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
7941  040502  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
7942  040510  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
7943  040516  000416                          BR      8$              ;GO ISSUE COMMAND
7944
7945  040520  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
7946  040526  001003                          BNE     7$              ;NO, CHECK IF RECAL
7947  040530  013737  003076  003124          MOV     @W.MIN,@W.DRV   ;LOAD WATCH DOG TIME FOR 1 MINUTE
7948  040536  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
7949  040544  001003                          BNE     8$              ;NO, CONTINUE
```

```
 7950  040546  013737  003074  003124          MOV     W.BSEC,W.DRV     ;LOAD RECAL TIME FOR 8 SECONDS
 7951  040554  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 7952  040562  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
 7953                                                                  ;  AND DRIVE TYPE
 7954  040570  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
 7955  040576  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
 7956  040604  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
 7957  040612  001533                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
 7958  040614  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
 7959  040622  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 7960  040630  004737  034572          10$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
 7961  040634  016237  000000  003010          MOV     RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REGISTER 1
 7962  040642  032737  000200  003010          BIT     #RDY,T.CS1       ;WAIT FOR READY
 7963  040650  001767                          BEQ     10$
 7964  040652  032737  100000  003010          BIT     #CERR,T.CS1      ;CHECK FOR ERROR
 7965  040660  001011                          BNE     15$              ;YES, GIVE NORMAL RETURN
 7966  040662  004737  034572          11$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
 7967  040666  016237  000016  003024          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 7968  040674  133737  003111  003025          BITB    INTMSK,T.ASOF+1  ;CHECK IF INTERRUPT HAS OCCURRED
 7969  040702  001767                          BEQ     11$              ;WAIT FOR DRIVE INTERRUPT
 7970  040704  105037  003110          15$:    CLRB    W.TIME           ;RESET TIMING ON THIS DRIVE
 7971  040710  005037  003124                  CLR     W.DRV            ;CLEAR DRIVE TIMING COUNT
 7972  040714  004737  040166                  JSR     PC,R.NORM        ;INDICATE COMMAND IS FINISHED
 7973  040720  000137  042074                  JMP     C.RTRN           ;RESTORE REGISTERS
 7974
 7975  040724  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
 7976  040732  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
 7977  040740  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
 7978  040746  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
 7979  040754  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
 7980  040762  001010                          BNE     25$              ;NO, GO ISSUE THE COMMAND
 7981  040764  032765  000200  000014          BIT     #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
 7982  040772  001404                          BEQ     25$              ;NO, GO ISSUE THE COMMAND
 7983  040774  012765  000123  000016          MOV     #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
 7984  041002  000406                          BR      26$              ;GO ISSUE COMMAND
 7985
 7986  041004  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
 7987  041012  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
 7988  041020  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 7989  041026  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
 7990                                                                  ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
 7991                                                                  ;  BITS 16-17
 7992  041034  010537  003064                  MOV     R5,O.WAIT        ;LOAD WAITING FOR COMMAND
 7993  041040  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
 7994  041046  001403                          BEQ     27$              ;NO, LOAD CS2
 7995  041050  052765  000020  000020          BIS     #BAI,P.CS2(R5)   ;SET INHIBIT BUS ADDRESS INCREMENT
 7996  041056  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
 7997  041064  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
 7998  041072  001403                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
 7999  041074  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
 8000  041102  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 8001  041110  000137  042074                  JMP     C.RTRN           ;RESTORE REGISTERS
 8002
 8003                                  .SBTTL   *SPECIAL COMMAND PROCESSING
 8004
 8005  041114  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
8006  041122  001132                        BNE     10$             ;NO, PROCESS OTHER COMMANDS
8007  041124  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
8008  041132  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
8009  041140  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
8010                                                                  ;  AND DRIVE TYPE
8011  041146  112765  000001  000016        MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
8012  041154  016562  000016  000000        MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
8013  041162  004737  034572          2$:   JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
8014  041166  016265  000000  000016        MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
8015  041174  032765  000200  000016        BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
8016  041202  001767                        BEQ     2$
8017  041204  004737  037644                JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
8018  041210  016265  000034  000040        MOV     RKMR2(R2),P.A00(R5) ;STORE STATUS BYTE 00 MESSAGE A
8019  041216  016265  000036  000042        MOV     RKMR3(R2),P.B00(R5) ;STORE STATUS BYTE 00 MESSAGE B
8020  041224  032765  100000  000016        BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
8021  041232  001436                        BEQ     6$              ;NO, GATHER DRIVE STATUS
8022  041234  105037  003110                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
8023  041240  005037  003124                CLR     W.DRV           ;CLEAR WATCH DOG COUNT
8024  041244  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8025  041252  001043                        BNE     8$              ;YES, INDICATE NORMAL RETURN
8026  041254  032765  001000  000020        BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
8027  041262  001043                        BNE     9$              ;YES, INDICATE CONTROLLER ERROR
8028  041264  004037  037170                JSR     R0,I.CCLR       ;CLEAR ERROR
8029  041270  042074                        C.RTRN                  ;ERROR RETURN
8030  041272  032765  010400  000020        BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
8031  041300  001007                        BNE     5$              ;REPORT ERROR
8032  041302  032765  000001  000036        BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
8033  041310  001003                        BNE     5$              ;YES, REPORT ERROR
8034  041312  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
8035  041320  004737  040154          5$:   JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
8036  041324  000137  042074                JMP     C.RTRN          ;RESTORE REGISTERS
8037
8038  041330  004037  037726          6$:   JSR     R0,I.STAT       ;GATHER DRIVE STATUS
8039  041334  042074                        C.RTRN                  ;ERROR RETURN
8040  041336  105037  003110                CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
8041  041342  005037  003124                CLR     W.DRV           ;RESET WATCH-DOG TIME
8042  041346  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8043  041354  001402                        BEQ     8$              ;NO, REPORT ERROR
8044  041356  005062  000000                CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
8045  041362  004737  040166          8$:   JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
8046  041366  000137  042074                JMP     C.RTRN          ;RESTORE REGISTERS
8047
8048  041372  052737  100000  003062  9$:   BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
8049  041400  004737  040200                JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
8050  041404  000137  042074                JMP     C.RTRN
8051
8052  041410  122765  000140  000001  10$:  CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
8053  041416  001040                        BNE     13$             ;NO, CHECK IF READ ALL HEADERS
8054  041420  010537  003064                MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
8055                                                                  ;  WAIT FOR COMMAND
8056  041424  052765  000010  000020        BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
8057  041432  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
8058  041440  112737  000001  003104        MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
8059  041446  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
8060  041454  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
8061                                                                  ;  AND DRIVE TYPE
```

```
8062  041462  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
8063  041470  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8064  041476  001403                          BEQ     11$               ;NO, DO NOT RESET INTERRUPT ENABLE
8065  041500  042765  000100  000016          BIC     #IE,P.CS1(R5)     ;RESET INTERRUPT ENABLE
8066  041506  016562  000016  000000  11$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
8067  041514  000137  042074                  JMP     C.RTRN            ;RESTORE REGISTERS
8068
8069  041520  122765  000164  000001  13$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
8070  041526  001053                          BNE     30$               ;NO, CHECK IF CONTROLLER CLEAR
8071  041530  010537  003064                  MOV     R5,O.WAIT         ;SET WAITING FOR COMMAND COMMPLETION
8072  041534  016537  000010  003100          MOV     P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
8073  041542  132765  000020  000007          BITB    #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
8074  041550  001404                          BEQ     14$               ;YES, LOAD 22 IN HEADER COUNT
8075  041552  012737  000024  003102          MOV     #20.,HDR.CT       ;LOAD 20 IN SECTOR COUNT
8076  041560  000403                          BR      22$               ;GO ISSUE READ HEADER COMMAND
8077
8078  041562  012737  000026  003102  14$:    MOV     #22.,HDR.CT       ;LOAD 22 IN SECTOR COUNT
8079  041570  016562  000002  000020  22$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
8080  041576  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
8081  041604  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
8082  041612  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
8083  041620  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
8084                                                  ;  AND FORMAT
8085  041626  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
8086  041634  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8087  041642  001027                          BNE     34$               ;YES, INDICATE ILLEGAL DRIVER COMMAND
8088  041644  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
8089  041652  000137  042074                  JMP     C.RTRN            ;RESTORE REGISTERS
8090
8091  041656  122765  000176  000001  30$:    CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
8092  041664  001012                          BNE     32$               ;NO, CHECK IF SUBSYSTEM CLEAR
8093  041666  004037  037170                  JSR     R0,I.CCLR         ;CLEAR CONTROLLER
8094  041672  042074                          C.RTRN                    ;ERROR RETURN
8095  041674  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8096  041702  001472                          BEQ     40$               ;NO, INDICATE NORMAL RETURN
8097  041704  005062  000000                  CLR     RKCS1(R2)         ;RESET INTERRUPT ENABLE
8098  041710  000467                          BR      40$               ;INDICATE NORMAL RETURN
8099
8100  041712  122765  000177  000001  32$:    CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
8101  041720  001406                          BEQ     36$               ;YES, CLEAR SUBSYSTEM
8102  041722  052737  000100  003062  34$:    BIS     #E.ILLD,E.CONT    ;SET ILLEGAL DRIVER COMMAND
8103  041730  004737  040200                  JSR     PC,R.CONT         ;REPORT ERROR
8104  041734  000457                          BR      C.RTRN            ;RESTORE REGISTERS
8105
8106  041736  012762  000040  000010  36$:    MOV     #SCLR,RKCS2(R2)   ;ISSUE SUBSYSTEM CLEAR
8107  041744  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
8108  041752  032765  100000  000016          BIT     #CERR,P.CS1(R5)   ;CLEAR IF CONTROLLER ERROR RESET
8109  041760  001406                          BEQ     37$               ;NO, FINISH COMMAND
8110  041762  052737  000001  003062          BIS     #BIT0,E.CONT      ;SET CLEAR SUBSYSTEM DID NOT CLEAR
8111                                                                    ;  CONTROLLER ERROR
8112  041770  004737  040200                  JSR     PC,R.CONT         ;REPORT ERROR
8113  041774  000437                          BR      C.RTRN            ;RESTORE REGISTERS
8114
8115  041776  013746  003070          37$:    MOV     W.MILI,-(SP)      ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
8116                                                                    ;  TO DISAPPEAR
8117  042002  016265  000000  000016  38$:    MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
8118   042010  032765  040000  000016          BIT     #DI,P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
8119   042016  001411                          BEQ     39$             ;YES, FINISH COMMAND
8120   042020  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
8121   042022  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
8122   042024  005726                          TST     (SP)+           ;ADJUST STACK
8123   042026  052737  000040  003062          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
8124                                                                   ;   DRIVE ATTENTIONS
8125   042034  004737  040200                  JSR     PC,R.CONT       ;REPORT ERROR
8126   042040  000415                          BR      C.RTRN          ;RESTORE REGISTER
8127
8128   042042  005726                  39$:    TST     (SP)+           ;ADJUST STACK
8129   042044  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
8130   042052  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
8131   042054  112737  177777  003104          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
8132   042062  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
8133   042070  004737  040166          40$:    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
8134
8135   042074  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
8136   042100  012600                          MOV     (SP)+,R0        ;RESTORE R0
8137   042102  012601                          MOV     (SP)+,R1        ;RESTORE R1
8138   042104  012602                          MOV     (SP)+,R2        ;RESTORE R2
8139   042106  012603                          MOV     (SP)+,R3        ;RESTORE R3
8140   042110  012604                          MOV     (SP)+,R4        ;RESTORE R4
8141   042112  012605                          MOV     (SP)+,R5        ;RESTORE R5
8142   042114  000207                          RTS     PC              ;RETURN
8143                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
8144
8145                                  ;;**************************************************************
8146                                  ;;*
8147                                  ;;*     THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
8148                                  ;;*     WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
8149                                  ;;*     IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
8150                                  ;;*     ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
8151                                  ;;*
8152                                  ;;*CALL
8153                                  ;;*     MOV     <ADDRESS OF ASCII STRING>,-(SP)
8154                                  ;;*     JSR     PC,OCTBIN
8155                                  ;;*     <ADDRESS OF ERROR RETURN>
8156                                  ;;*     RETURN
8157                                  ;;*
8158                                  ;;**************************************************************
8159
8160   042116  010046          OCTBIN: MOV     R0,-(SP)        ;SAVE R0
8161   042120  010146                  MOV     R1,-(SP)        ;SAVE R1
8162   042122  010246                  MOV     R2,-(SP)        ;SAVE R2
8163   042124  016600  000010          MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
8164   042130  005001                  CLR     R1              ;CLEAR DATA WORDS
8165   042132  005002                  CLR     R2
8166   042134  112046          2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
8167   042136  001423                  BEQ     3$              ;IF ZERO GET OUT
8168   042140  121627  000054          CMPB    (SP),#',        ;CHECK IF COMMA
8169   042144  001420                  BEQ     3$              ;IF COMMA GET OUT
8170   042146  122716  000060          CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
8171   042152  003030                  BGT     4$              ;   AN OCTAL DIGIT
8172   042154  122716  000067          CMPB    #'7,(SP)
8173   042160  002425                  BLT     4$
```

```
8174  042162  006301                        ASL     R1              ; *2
8175  042164  006102                        ROL     R2
8176  042166  006301                        ASL     R1              ; *4
8177  042170  006102                        ROL     R2
8178  042172  006301                        ASL     R1              ; *8
8179  042174  006102                        ROL     R2
8180  042176  042716  177770              BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
8181  042202  062601                        ADD     (SP)+,R1        ;ADD THIS DIGIT
8182  042204  000753                        BR      2$              ;LOOP
8183  042206  005726                3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
8184  042210  010166  000010              MOV     R1,10(SP)       ;SAVE RESULT
8185  042214  010237  042250              MOV     R2,$HIOCT
8186  042220  012602                        MOV     (SP)+,R2        ;RESTORE R2
8187  042222  012601                        MOV     (SP)+,R1        ;RESTORE R1
8188  042224  012600                        MOV     (SP)+,R0        ;RESTORE R0
8189  042226  062716  000002              ADD     #2,(SP)         ;ADJUST RETURN
8190  042232  000207                        RTS     PC              ;RETURN
8191
8192  042234  005726                4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
8193  042236  012602                        MOV     (SP)+,R2        ;RESTORE R2
8194  042240  012601                        MOV     (SP)+,R1        ;RESTORE R1
8195  042242  012600                        MOV     (SP)+,R0        ;RESTORE R0
8196  042244  013616                        MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
8197  042246  000207                        RTS     PC              ;GO PROCESS ERROR
8198  042250  000000                $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
8199                                 .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
8200
8201                                 ;;**************************************************************
8202                                 ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
8203                                 ;*WITH A RANGE OF 0 TO 2(+33)-1.
8204                                 ;*CALL:
8205                                 ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
8206                                 ;*      RETURN                  ;;RETURN HERE THE RANDOM
8207                                 ;*                              ;;NUMBER WILL BE IN
8208                                 ;*                              ;;$HINUM,$LONUM
8209
8210  042252                         $RAND:
8211  042252  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8212  042254  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8213  042256  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8214  042260  013700  042352              MOV     $LONUM,R0       ;;SET R0 WITH LOW
8215  042264  013701  042350              MOV     $HINUM,R1       ;;SET R1 WITH HIGH
8216  042270  012702  177771              MOV     #-7,R2          ;;SET SHIFT COUNT
8217  042274  006300                1$:     ASL     R0              ;;SHIFT R0 LEFT AND
8218  042276  006101                        ROL     R1              ;;ROTATE CARRY INTO R1 AND
8219  042300  005202                        INC     R2              ;;CHECK FOR DONE
8220  042302  001374                        BNE     1$              ;;CONTINUE SHIFT LOOP
8221  042304  063700  042352              ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
8222  042310  005501                        ADC     R1              ;;PROPOGATE CARRY
8223  042312  063701  042350              ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
8224  042316  062700  001057              ADD     #1057,R0        ;;ADD LOW CONSTANT
8225  042322  005501                        ADC     R1              ;;PROPOGATE CARRY
8226  042324  062701  047401              ADD     #47401,R1       ;;ADD HIGH CONSTANT
8227  042330  010037  042352              MOV     R0,$LONUM       ;;SAVE R0
8228  042334  010137  042350              MOV     R1,$HINUM       ;;SAVE R1
8229  042340  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
8230  042342  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8231  042344  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8232  042346  000207                  RTS     PC              ;;RETURN
8233  042350  176543          $HINUM: .WORD   176543
8234  042352  123456          $LONUM: .WORD   123456
8235                           .SBTTL  TYPE ROUTINE
8236
8237                           ;;*****************************************************************
8238                           ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8239                           ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8240                           ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
8241                           ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
8242                           ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
8243                           ;*
8244                           ;*CALL:
8245                           ;*1) USING A TRAP INSTRUCTION
8246                           ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
8247                           ;*OR
8248                           ;*      TYPE
8249                           ;*      MESADR
8250                           ;*
8251
8252  042354  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
8253  042360  100002                  BPL     1$              ;;BR IF YES
8254  042362  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
8255  042364  000430                  BR      3$              ;;LEAVE
8256  042366  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
8257  042370  017600  000002          MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
8258  042374  122737  000001  001350  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
8259  042402  001011                  BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
8260  042404  132737  000100  001351  BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
8261  042412  001405                  BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
8262  042414  010037  042424          MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
8263  042420  004737  044764          JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
8264  042424  000000          61$:    .WORD   0               ;;MESSAGE ADDRESS
8265  042426  132737  000040  001351  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
8266  042434  001003                  BNE     60$             ;;YES,SKIP TYPE OUT
8267  042436  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
8268  042440  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
8269  042442  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
8270  042444  012600          60$:    MOV     (SP)+,R0        ;;RESTORE R0
8271  042446  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
8272  042452  000002                  RTI                     ;;RETURN
8273  042454  122716  000011  4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
8274  042460  001430                  BEQ     8$
8275  042462  122716  000200          CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
8276  042466  001006                  BNE     5$
8277  042470  005726                  TST     (SP)+           ;;POP  <CR><LF> EQUIV
8278  042472  104401                  TYPE                    ;;TYPE A CR AND LF
8279  042474  001325                  $CRLF
8280  042476  105037  042632          CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
8281  042502  000755                  BR      2$              ;;GET NEXT CHARACTER
8282  042504  004737  042566  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
8283  042510  123726  001156  6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
8284  042514  001350                  BNE     2$              ;;IF NO GO GET NEXT CHAR.
8285  042516  013746  001154          MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
8286
8287  042522  105366  000001          7$:     DECB    1(SP)           ;;AND THE NULL CHAR.
8288  042526  002770                          BLT     6$              ;;DOES A NULL NEED TO BE TYPED?
8289  042530  004737  042566                  JSR     PC,$TYPEC       ;;BR IF NO--GO POP THE NULL OFF OF STACK
8290  042534  105337  042632                  DECB    $CHARCNT        ;;GO TYPE A NULL
8291  042540  000770                          BR      7$              ;;DO NOT COUNT AS A COUNT
8292                                                                  ;;LOOP
8293                                  ;HORIZONTAL TAB PROCESSOR
8294
8295  042542  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
8296  042546  004737  042566          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
8297  042552  132737  000007  042632          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
8298  042560  001372                          BNE     9$              ;;TAB STOP
8299  042562  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
8300  042564  000724                          BR      2$              ;;GET NEXT CHARACTER
8301  042566  105777  136356          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
8302  042572  100375                          BPL     $TYPEC
8303  042574  116677  000002  136350          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8304  042602  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
8305  042610  001003                          BNE     1$              ;;BRANCH IF NO
8306  042612  105037  042632                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
8307  042616  000406                          BR      $TYPEX          ;;EXIT
8308  042620  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
8309  042626  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
8310  042630  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
8311  042632  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
8312  042634  000207          $TYPEX: RTS     PC
8313
8314
8315
8316
8317                          ;;*****************************************************************
8318                          .SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
8319                          ;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
8320                          ;*      USES ALL REGISTERS (R0-R5)
8321                          ;*
8322                          ;*      ENTER WITH   JSR   PC,M.DPIM
8323                          ;*                   MULTIPLIER IN R2-R3
8324                          ;*                   MULTIPLICAND IN R4-R5
8325                          ;*                   PRODUCT RETURNED IN R0-R1-R2-R3
8326                          ;*
8327                          ;*****************************************************************
8328  042636  005000          M.DPIM: CLR     R0              ;CLEAR HI ORDER WORDS
8329  042640  005001                  CLR     R1
8330  042642  012746  000041          MOV     #41,-(SP)       ;MOVE 33 (DEC) TO COUNTER
8331  042646  006000          M.DP01: ROR     R0
8332  042650  006001                  ROR     R1
8333  042652  006002                  ROR     R2              ;SHIFT TO ADD
8334  042654  006003                  ROR     R3
8335  042656  103003                  BCC     M.DP02          ;NO CARRY NO ADD
8336  042660  060501                  ADD     R5,R1
8337  042662  005500                  ADC     R0              ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
8338  042664  060400                  ADD     R4,R0           ;  PRODUCT
8339  042666  005316          M.DP02: DEC     @SP             ;DECREMENT COUNTER
8340  042670  001366                  BNE     M.DP01
8341  042672  005726                  TST     (SP)+           ;REMOVE THE COUNTER
```

```
 8342  042674  000207                        RTS     PC
 8343
 8344
 8345                                  ;;*****************************************************************
 8346                                  .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
 8347                                  ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
 8348                                  ;*              USES ALL REGISTERS (R0-R5)
 8349                                  ;*
 8350                                  ;*              ENTER WITH  JSR   PC,M.DPID
 8351                                  ;*                      DIVIDEND IN R0-R1-R2-R3
 8352                                  ;*                      DIVISOR IN R4-R5
 8353                                  ;*                      REMAINDER RETURNED IN R0-R1
 8354                                  ;*                      QUOTIENT RETURNED IN R2-R3
 8355                                  ;*
 8356                                  ;;*****************************************************************
 8357  042676  012746  000040         M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
 8358  042702  010446                         MOV     R4,-(SP)        ;HI ORDER
 8359  042704  010546                         MOV     R5,-(SP)        ;LO ORDER DIVISOR TO THE STACK
 8360  042706  005466  000002                 NEG     2(SP)           ;FORM NEGATIVE
 8361  042712  005416                         NEG     @SP             ;   VERSION OF THE DIVISOR
 8362  042714  005666  000002                 SBC     2(SP)
 8363  042720  061601                         ADD     @SP,R1
 8364  042722  005500                         ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
 8365  042724  066600  000002                 ADD     2(SP),R0
 8366  042730  103445                         BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
 8367  042732  005046                         CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
 8368  042734  006103                 M.DP40: ROL     R3
 8369  042736  006102                         ROL     R2
 8370  042740  006101                         ROL     R1
 8371  042742  006100                         ROL     R0
 8372  042744  005716                         TST     @SP             ;TEST "CARRY INDICATOR"
 8373  042746  001410                         BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
 8374  042750  005016                         CLR     @SP             ;CLEAR UP FOR NEXT TIME
 8375  042752  066601  000002                 ADD     2(SP),R1
 8376  042756  005500                         ADC     R0              ;ADD -(DIVISOR)
 8377  042760  005516                         ADC     @SP     ; I     ;SET "CARRY"
 8378  042762  066600  000004                 ADD     4(SP),R0 ;<
 8379  042766  000404                         BR      M.DP42
 8380  042770  060501                 M.DP41: ADD     R5,R1
 8381  042772  005500                         ADC     R0              ;ADD +(DIVISOR)
 8382  042774  005516                         ADC     @SP     ; I     ;SET "CARRY"
 8383  042776  060400                         ADD     R4,R0   ;<
 8384  043000  005516                 M.DP42: ADC     @SP             ;SET "CARRY"
 8385  043002  005716                         TST     @SP             ;TEST THE UPDATE INDICATOR
 8386  043004  001401                         BEQ     .+4     ;>      ;IF ZERO FORGET IT
 8387  043006  005203                         INC     R3      ; I     ;NO CARRY POSSIBLE HERE
 8388  043010  005366  000006                 DEC     6(SP)   ;<      ;DECREMENT COUNTER
 8389  043014  003347                         BGT     M.DP40          ;BR IF MORE TO DO
 8390  043016  006003                         ROR     R3
 8391  043020  103404                         BCS     M.DP44
 8392  043022  060501                         ADD     R5,R1
 8393  043024  005500                         ADC     R0
 8394  043026  060400                         ADD     R4,R0
 8395  043030  000241                         CLC
 8396  043032  006103                 M.DP44: ROL     R3
 8397  043034  062706  000010                 ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
```

```
8398   043040  000242                          CLV
8399   043042  000207                          RTS     PC
8400   043044  062706  000006          M.DP50: ADD     #6,SP
8401   043050  000262                          SEV
8402   043052  000207                          RTS     PC
8403
8404
8405
8406                                   .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
8407
8408                                   ;;**********************************************************************
8409                                   ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
8410                                   ;*UNSIGNED OCTAL ASCIZ NUMBER.
8411                                   ;*CALL
8412                                   ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
8413                                   ;*      JSR     PC,@#$DB20      ;;CALL THE ROUTINE
8414                                   ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
8415
8416
8417   043054  104407                  $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
8418   043056  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
8419   043062  012705  043173                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
8420   043066  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
8421   043072  012703  177770                  MOV     #^C7,R3         ;;MASK
8422   043076  012100                          MOV     (R1)+,R0        ;;LOWER WORD
8423   043100  012101                          MOV     (R1)+,R1        ;;HIGH WORD
8424   043102  005002                          CLR     R2              ;;TERMINATOR
8425   043104  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
8426   043106  010002                          MOV     R0,R2           ;;GET THIS DIGIT
8427   043110  005304                          DEC     R4              ;;COUNT THIS CHARACTER
8428   043112  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
8429   043114  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
8430   043116  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
8431   043120  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
8432   043124  104410                          RESREG                  ;;RESTORE ALL REGISTERS
8433   043126  000207                          RTS     PC              ;;RETURN TO USER
8434   043130  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
8435   043132  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
8436   043134  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
8437   043136  006001                          ROR     R1
8438   043140  006000                          ROR     R0
8439   043142  006001                          ROR     R1
8440   043144  006000                          ROR     R0
8441   043146  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
8442   043150  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
8443   043154  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
8444   043156  000016          $OCTVL: .BLKB   14.                     ;;RESERVE DATA TABLE
8445                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
8446
8447                                   ;;**********************************************************************
8448                                   ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
8449                                   ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
8450                                   ;*POSITIVE.
8451                                   ;*CALL
8452                                   ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
8453                                   ;*      JSR     PC,@#$DB2D
```

```
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK
                                        $DBED:  SAVREG                  ;;SAVE REGISTERS
043174  104407                                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
043176  016602  000002                          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
043202  012700  043354                          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
043206  010066  000002                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
043212  012201                                  MOV     (R2)+,R2
043214  012202
043216  012737  000012  043272                  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
043224  012704  043304                          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
043230  012705  043306                          MOV     #$TNPWR+2,R5
043234  005003                          1$:     CLR     R3              ;;CLEAR PARTIAL
043236  161401                          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
043240  005602                                  SBC     R2
043242  161502                                  SUB     (R5),R2
043244  002402                                  BLT     3$              ;;BR IF TEN POWER TO LARGE
043246  005203                                  INC     R3              ;;ADD 1 TO PARTIAL
043250  000772                                  BR      2$              ;;LOOP
043252  062401                          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
043254  005502                                  ADC     R2
043256  062402                                  ADD     (R4)+,R2
043260  022525                                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
043262  052703  000060                          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
043266  110320                                  MOVB    R3,(R0)+        ;;SAVE IT
043270  005327                                  DEC     (PC)+           ;;DONE?
043272  000000                          4$:     .WORD   0
043274  001357                                  BNE     1$              ;;BR IF NO
043276  105020                                  CLRB    (R0)+           ;;TERMINATOR
043300  104410                                  RESREG                  ;;RESTORE REGISTERS
043302  000207                                  RTS     PC              ;;RETURN
043304  145000                          $TNPWR: 145000                  ;;1.0E09
043306  035632                                  35632
043310  160400                                  160400                  ;;1.0E08
043312  002765                                  2765
043314  113200                                  113200                  ;;1.0E07
043316  000230                                  230
043320  041100                                  041100                  ;;1.0E06
043322  000017                                  17
043324  103240                                  103240                  ;;1.0E05
043326  000001                                  1
043330  023420                                  23420                   ;;1.0E04
043332  000000                                  0
043334  001750                                  1750                    ;;1.0E03
043336  000000                                  0
043340  000144                                  144                     ;;1.0E02
043342  000000                                  0
043344  000012                                  12                      ;;1.0E01
043346  000000                                  0
043350  000001                                  1                       ;;1.0E00
043352  000000                                  0
043354  000014                          $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
                                        .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

```
                                        TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

043370  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
043372  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
043376  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
043400  001403                          BEQ     2$              ;;BR IF YES
043402  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
043406  001773                          BEQ     1$              ;;BR IF YES
043410  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
043412  010037  043420                  MOV     R0,3$           ;;SAVE FOR TYPING
043416  104401                          TYPE                    ;;GO TYPE
043420  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
043422  012600                          MOV     (SP)+,R0        ;;RESTORE R0
043424  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
043426  000207                          RTS     PC              ;;RETURN
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT

043430  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
043434  116637  000001  043653          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
043442  112637  043655                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
043446  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
043452  000406                          BR      $TYPON
043454  112737  000001  043653  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
043462  112737  000006  043655          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
043470  112737  000005  043652  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
043476  010346                          MOV     R3,-(SP)        ;;SAVE R3
043500  010446                          MOV     R4,-(SP)        ;;SAVE R4
043502  010546                          MOV     R5,-(SP)        ;;SAVE R5
```

```
8566   043504   113704   043655                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
8567   043510   005404                          NEG     R4
8568   043512   062704   000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
8569   043516   110437   043654                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
8570   043522   113704   043653                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
8571   043526   016605   000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
8572   043532   005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
8573   043534   006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
8574   043536   000404                          BR      3$              ;;GO DO MSB
8575   043540   006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
8576   043542   006105                          ROL     R5
8577   043544   006105                          ROL     R5
8578   043546   010503                          MOV     R5,R3
8579   043550   006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
8580   043552   105337   043654                 DECB    $OMODE          ;;TYPE THIS DIGIT?
8581   043556   100016                          BPL     7$              ;;BR IF NO
8582   043560   042703   177770                 BIC     #177770,R3      ;;GET RID OF JUNK
8583   043564   001002                          BNE     4$              ;;TEST FOR 0
8584   043566   005704                          TST     R4              ;;SUPPRESS THIS 0?
8585   043570   001403                          BEQ     5$              ;;BR IF YES
8586   043572   005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
8587   043574   052703   000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
8588   043600   052703   000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
8589   043604   110337   043650                 MOVB    R3,8$           ;;SAVE FOR TYPING
8590   043610   104401   043650                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
8591   043614   105337   043652         7$:     DECB    $OCNT           ;;COUNT BY 1
8592   043620   003347                          BGT     2$              ;;BR IF MORE TO DO
8593   043622   002402                          BLT     6$              ;;BR IF DONE
8594   043624   005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
8595   043626   000744                          BR      2$              ;;GO DO THE LAST DIGIT
8596   043630   012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
8597   043632   012604                          MOV     (SP)+,R4        ;;RESTORE R4
8598   043634   012603                          MOV     (SP)+,R3        ;;RESTORE R3
8599   043636   016666   000002   000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
8600   043644   012616                          MOV     (SP)+,(SP)
8601   043646   000002                          RTI                     ;;RETURN
8602   043650      000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
8603   043651      000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
8604   043652      000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
8605   043653      000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
8606   043654   000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
8607                                   .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
8608
8609                                   ;;*****************************************************************
8610                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
8611                                   ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
8612                                   ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
8613                                   ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
8614                                   ;*REPLACED WITH SPACES.
8615                                   ;*CALL:
8616                                   ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
8617                                   ;*      TYPDS                   ;;GO TO THE ROUTINE
8618
8619   043656                          $TYPDS:
8620   043656   010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8621   043660   010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
8622   043662  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8623   043664  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
8624   043666  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
8625   043670  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
8626   043674  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
8627   043700  100004                          BPL     1$              ;;BR IF INPUT IS POS.
8628   043702  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
8629   043704  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
8630   043712  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
8631   043714  012703  044072                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
8632   043720  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
8633   043724  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
8634   043726  016001  044062                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
8635   043732  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
8636   043734  002402                          BLT     4$              ;;BR IF DONE
8637   043736  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
8638   043740  000774                          BR      3$
8639   043742  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
8640   043744  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
8641   043746  001002                          BNE     5$              ;;FALL THROUGH IF 0
8642   043750  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
8643   043752  100407                          BMI     7$              ;;BR IF YES
8644   043754  106316                  5$:     ASLB    (SP)            ;;MSD?
8645   043756  103003                          BCC     6$              ;;BR IF NO
8646   043760  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
8647   043766  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
8648   043772  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
8649   043776  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
8650   044000  005720                          TST     (R0)+           ;;JUST INCREMENTING
8651   044002  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
8652   044006  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
8653   044010  003002                          BGT     8$              ;;GO TO EXIT
8654   044012  010502                          MOV     R5,R2           ;;GET THE LSD
8655   044014  000764                          BR      6$              ;;GO CHANGE TO ASCII
8656   044016  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
8657   044020  100003                          BPL     9$              ;;BR IF NO
8658   044022  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
8659   044030  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
8660   044032  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
8661   044034  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
8662   044036  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
8663   044040  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
8664   044042  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
8665   044044  104401  044072                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
8666   044050  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
8667   044056  012616                          MOV     (SP)+,(SP)
8668   044060  000002                          RTI                     ;;RETURN TO USER
8669   044062  023420                  $DTBL:  10000.
8670   044064  001750                          1000.
8671   044066  000144                          100.
8672   044070  000012                          10.
8673   044072  000004                  $DBLK:  .BLKW   4
8674                                   .SBTTL  ERROR HANDLER ROUTINE
8675
8676                                   ;;**********************************************************************
8677                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

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

044102                                  $ERROR:
044102  105237  001103                  7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
044106  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
044110  013777  001102  135024          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
044116  032777  002000  135014          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
044124  001402                          BEQ     1$              ;;NO - SKIP
044126  104401  001320                  TYPE    ,$BELL          ;;RING BELL
044132  005237  001112                  1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
044136  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
044142  162737  000002  001116          SUB     #2,$ERRPC
044150  117737  134742  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
044156  032777  020000  134754          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
044164  001004                          BNE     20$             ;;SKIP TYPEOUTS
044166  004737  030352                  JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
044172  104401  001325                  TYPE    ,$CRLF
044176                                  20$:
044176  122737  000001  001350          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
044204  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
044206  113737  001114  044220          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
044214  004737  044774                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
044220     000                          21$:    .BYTE   0
044221     000                          .BYTE   0
044222  000777                          22$:    BR      22$             ;;APT ERROR LOOP
044224  005777  134710                  2$:     TST     @SWR            ;;HALT ON ERROR
044230  100001                          BPL     3$              ;;SKIP IF CONTINUE
044232  000000                          HALT                    ;;HALT ON ERROR!
044234  032777  001000  134676          3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
044242  001402                          BEQ     4$              ;;BR IF NO
044244  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
044250  005737  001316                  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
044254  001402                          BEQ     5$              ;;BR IF NONE
044256  013716  001316                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
044262                                  5$:
044262  022737  017264  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
044270  001001                          BNE     6$              ;;BRANCH IF NO
044272  000000                          HALT                    ;;YES
044274                                  6$:
044274  000002                          RTI                     ;;RETURN
                                        .SBTTL  TTY INPUT ROUTINE

                                        ;;**************************************************************
                                        .ENABL  LSB

                                        .DSABL  LSB
```

```
8734                                    ;;*****************************************************************
8735                                    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8736                                    ;*CALL:
8737                                    ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
8738                                    ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
8739                                    ;*                              ;;WITH PARITY BIT STRIPPED OFF
8740                                    ;
8741
8742  044276  011646                    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
8743  044300  016666  000004  000002            MOV     4(SP),2(SP)     ;;SAVE THE PS
8744  044306  105777  134632            1$:     TSTB    @$TKS           ;;WAIT FOR
8745  044312  100375                            BPL     1$              ;;A CHARACTER
8746  044314  117766  134626  000004            MOVB    @$TKB,4(SP)     ;;READ THE TTY
8747  044322  042766  177600  000004            BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
8748  044330  026627  000004  000023            CMP     4(SP),#23       ;;IS IT A CONTROL-S?
8749  044336  001013                            BNE     3$              ;;BRANCH IF NO
8750  044340  105777  134600            2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
8751  044344  100375                            BPL     2$              ;;LOOP UNTIL ITS THERE
8752  044346  117746  134574                    MOVB    @$TKB,-(SP)     ;;GET CHARACTER
8753  044352  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
8754  044356  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
8755  044362  001366                            BNE     2$              ;;IF NOT DISCARD IT
8756  044364  000750                            BR      1$              ;;YES, RESUME
8757  044366  026627  000004  000140    3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
8758  044374  002407                            BLT     4$              ;;BRANCH IF YES
8759  044376  026627  000004  000175            CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
8760  044404  003003                            BGT     4$              ;;BRANCH IF YES
8761  044406  042766  000040  000004            BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
8762  044414  000002                    4$:     RTI                     ;;GO BACK TO USER
8763  044416  052536  005015     000    $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
8764  044423     136  006507  000012    $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
8765  044430  005015  053523  020122    $MSWR:  .ASCIZ  <15><12>/SWR = /
8766  044436  020075     000
8767  044441     040  047040  053505    $MNEW:  .ASCIZ  /  NEW = /
8768  044446  036440  000040
8769
8770
8771
8772                                            .SBTTL  POWER DOWN AND UP ROUTINES
8773
8774                                    ;POWER DOWN ROUTINE
8775  044452  012737  044464  000024    $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
8776  044460  000000                            HALT                    ;HANG UP
8777  044462  000776                            BR      .-2
8778
8779                                    ;POWER UP ROUTINE
8780  044464  005037  044536            $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
8781  044470  005237  044536            4$:     INC     $PWRCT
8782  044474  001375                            BNE     4$
8783  044476  012737  044452  000024            MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
8784  044504  012737  000340  000026            MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
8785  044512  012737  000340  000036            MOV     #PR7,TRAPVEC+2
8786  044520  012706  001100                    MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
8787  044524  104401  044540                    TYPE    ,PWRMSG         ;TYPE "POWER FAILED"
8788  044530  000005                            RESET                   ;CLEAR THE UNIBUS
8789  044532  000177  134350                    JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
8790  044536  000000                  $PWRCT: .WORD   0
8791  044540  005015  047520  042527  PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
8792  044546  020122  040506  046111
8793  044554  042105  005015     000
8794          044562                  .EVEN
8795
8796
8797
8798                                  ;;*****************************************************************
8799                                  ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
8800                                  ;*CALLED BY "SCOPER"
8801                                  ;;*****************************************************************
8802  044562  105737  001103          SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
8803  044566  001406                          BEQ     6$              ;BR IF NOT
8804  044570  032777  001000  134342          BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
8805  044576  001402                          BEQ     6$              ;BR IF NOT
8806  044600  013716  001110                  MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
8807  044604  000002                  6$:     RTI                     ;RETURN
8808
8809
8810
8811                                  .SBTTL  SCOPE HANDLER ROUTINE
8812
8813                                  ;;*****************************************************************
8814                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
8815                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
8816                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
8817                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
8818                                  ;*SW14=1         LOOP ON TEST
8819                                  ;*SW09=1         LOOP ON ERROR
8820                                  ;*CALL
8821                                  ;*      SCOPE           ;;SCOPE=IOT
8822
8823  044606                          $SCOPE:
8824  044606  032777  040000  134324  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
8825  044614  001052                          BNE     $OVER           ;;YES IF SW14=1
8826                                  ;*****START OF CODE FOR THE XOR TESTER*****
8827  044616  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
8828                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
8829  044620  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
8830  044624  012737  044644  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
8831  044632  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
8832  044636  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
8833  044642  000421                          BR      $SVLAD          ;;GO TO THE NEXT TEST
8834  044644  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
8835  044646  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
8836  044652  000407                          BR      7$              ;;LOOP ON THE PRESENT TEST
8837  044654                          6$:;*****END OF CODE FOR THE XOR TESTER*****
8838  044654  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
8839  044660  001412                          BEQ     $SVLAD          ;;BR IF NO
8840  044662  032777  001000  134250          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
8841  044670  001404                          BEQ     4$              ;;BR IF NO
8842  044672  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
8843  044700  000420                          BR      $OVER
8844  044702  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
8845  044706  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
 8846  044712  113737  001102  001334          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 8847  044720  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 8848  044724  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 8849  044730  005037  001316                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 8850  044734  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 8851  044742  013777  001102  134172  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 8852  044750  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 8853  044754  000002                          RTI                     ;;FIXES PS
 8854                                  .SBTTL  APT COMMUNICATIONS ROUTINE
 8855
 8856                                  ;;**************************************************************************
 8857  044756  112737  000001  045222  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
 8858  044764  112737  000001  045220  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
 8859  044772  000403                          BR      $ATYC
 8860  044774  112737  000001  045222  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
 8861  045002                          $ATYC:
 8862  045002  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 8863  045004  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 8864  045006  105737  045220                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
 8865  045012  001450                          BEQ     5$              ;;IF NOT:  BR
 8866  045014  122737  000001  001350          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
 8867  045022  001031                          BNE     3$              ;;IF NOT:  BR
 8868  045024  132737  000100  001351          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 8869  045032  001425                          BEQ     3$              ;;IF NOT:  BR
 8870  045034  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
 8871  045040  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 8872  045046  005737  001330          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
 8873  045052  001375                          BNE     1$              ;;IF NOT:  WAIT
 8874  045054  010037  001344                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
 8875  045060  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
 8876  045062  001376                          BNE     2$
 8877  045064  163700  001344                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
 8878  045070  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
 8879  045072  010037  001346                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
 8880  045076  012737  000004  001330          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
 8881  045104  000413                          BR      5$
 8882  045106  017637  000004  045132  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
 8883  045114  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 8884  045122  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 8885  045126  004737  042354                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
 8886  045132  000000                  4$:     .WORD   0
 8887  045134                          5$:
 8888  045134  105737  045222          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 8889  045140  001416                          BEQ     12$             ;;IF NOT:  BR
 8890  045142  005737  001350                  TST     $ENV            ;;RUNNING UNDER APT?
 8891  045146  001413                          BEQ     12$             ;;IF NOT:  BR
 8892  045150  005737  001330          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 8893  045154  001375                          BNE     11$             ;;IF NOT:  WAIT
 8894  045156  017637  000004  001332          MOV     @4(SP),$FATAL   ;;GET ERROR #
 8895  045164  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 8896  045172  005237  001330                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 8897  045176  105037  045222          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 8898  045202  105037  045221                  CLRB    $LFLG           ;;CLEAR LOG FLAG
 8899  045206  105037  045220                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 8900  045212  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 8901  045214  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
 8902  045216  000207                          RTS     PC              ;;RETURN
 8903  045220     000  $MFLG:  .BYTE   0       ;;MESSG. FLAG
 8904  045221     000  $LFLG:  .BYTE   0       ;;LOG FLAG
 8905  045222     000  $FFLG:  .BYTE   0       ;;FATAL FLAG
 8906          045224          .EVEN
 8907          000200  APTSIZE=200
 8908          000001  APTENV=001
 8909          000100  APTSPOOL=100
 8910          000040  APTCSUP=040
 8911                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
 8912
 8913                  ;;**************************************************************
 8914                  ;*SAVE R0-R5
 8915                  ;*CALL:
 8916                  ;*      SAVREG
 8917                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
 8918                  ;*
 8919                  ;*TOP---(+16)
 8920                  ;* +2---(+18)
 8921                  ;* +4---R5
 8922                  ;* +6---R4
 8923                  ;* +8---R3
 8924                  ;*+10---R2
 8925                  ;*+12---R1
 8926                  ;*+14---R0
 8927
 8928  045224  $SAVREG:
 8929  045224  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 8930  045226  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 8931  045230  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 8932  045232  010346          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 8933  045234  010446          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 8934  045236  010546          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 8935  045240  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
 8936  045244  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
 8937  045250  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
 8938  045254  016646  000022  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
 8939  045260  000002          RTI
 8940
 8941                  ;*RESTORE R0-R5
 8942                  ;*CALL:
 8943                  ;*      RESREG
 8944  045262  $RESREG:
 8945  045262  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
 8946  045266  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
 8947  045272  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
 8948  045276  012666  000022  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
 8949  045302  012605          MOV     (SP)+,R5        ;;POP STACK INTO R5
 8950  045304  012604          MOV     (SP)+,R4        ;;POP STACK INTO R4
 8951  045306  012603          MOV     (SP)+,R3        ;;POP STACK INTO R3
 8952  045310  012602          MOV     (SP)+,R2        ;;POP STACK INTO R2
 8953  045312  012601          MOV     (SP)+,R1        ;;POP STACK INTO R1
 8954  045314  012600          MOV     (SP)+,R0        ;;POP STACK INTO R0
 8955  045316  000002          RTI
 8956                  .SBTTL  TRAP DECODER
 8957
```

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

045320  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
045322  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
045326  005740                          TST     -(R0)           ;;BACKUP BY 2
045330  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
045332  006300                          ASL     R0              ;;POSITION FOR INDEXING
045334  016000  045354                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
045340  000200                          RTS     R0              ;;GO TO ROUTINE


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

045342  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
045344  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
045352  000002                          RTI                     ;;RESTORE THE PSW


                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
045354  045342                  $TRPAD: .WORD   $TRAP2
045356  042354                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
045360  043454                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
045362  043430                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
045364  043470                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
045366  043656                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)


045370  044276                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
045372  045224                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
045374  045262                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
045376  044562                          SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE


045400  020040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
045406  052123  000040
045412  025052  020040     000  AS2SP2: .ASCIZ  /**  /
045417     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PARITY ERROR/
045424  020123  040520  044522
045432  054524  042440  051122
045440  051117     000
045443     116  047117  042455  EM2:    .ASCIZ  /NON-EXISTANT MEMORY/
045450  044530  052123  047101
045456  020124  042515  047515
045464  054522     000
045467     116  047117  042455  EM3:    .ASCIZ  /NON-EXISTANT DRIVE/
045474  044530  052123  047101
045502  020124  051104  053111
045510  000105
```

```
9014  045512  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERROR/
9015  045520  042511  042114  042440
9016  045526  051122  051117     000
9017  045533     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
9018  045540  020123  044524  042515
9019  045546  052517  000124
9020  045552  020104  047524  041440  EM6:    .ASCIZ  /D TO C PARITY ERROR/
9021  045560  050040  051101  052111
9022  045566  020131  051105  047522
9023  045574  000122
9024  045576  051104  053111  020105  EM7:    .ASCIZ  /DRIVE DETECTED PARITY ERROR/
9025  045604  042504  042524  052103
9026  045612  042105  050040  051101
9027  045620  052111  020131  051105
9028  045626  047522  000122
9029  045632  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
9030  045640     000
9031  045641     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
9032  045646  046040  051517  000123
9033  045654  046111  042514  040507  EM12:   .ASCIZ  /ILLEGAL FUNCTION/
9034  045662  020114  052506  041516
9035  045670  044524  047117     000
9036  045675     120  047522  051107  EM13:   .ASCIZ  /PROGRAMMING ERROR/
9037  045702  046501  044515  043516
9038  045710  042440  051122  051117
9039  045716     000
9040  045717     116  047117  042455  EM14:   .ASCIZ  /NON-EXISTANT FUNCTION/
9041  045724  044530  052123  047101
9042  045732  020124  052506  041516
9043  045740  044524  047117     000
9044  045745     104  044522  042526  EM15:   .ASCIZ  /DRIVE TYPE ERROR/
9045  045752  052040  050131  020105
9046  045760  051105  047522  000122
9047  045766  047506  046522  052101  EM16:   .ASCIZ  /FORMAT ERROR/
9048  045774  042440  051122  051117
9049  046002     000
9050  046003     127  044522  042524  EM17:   .ASCIZ  /WRITE LOCK ERROR/
9051  046010  046040  041517  020113
9052  046016  051105  047522  000122
9053  046024  051104  053111  020105  EM20:   .ASCIZ  /DRIVE UNSAFE/
9054  046032  047125  040523  042506
9055  046040     000
9056  046041     123  042505  020113  EM21:   .ASCIZ  /SEEK INCOMPLETE/
9057  046046  047111  047503  050115
9058  046054  042514  042524     000
9059  046061     103  046131  047111  EM22:   .ASCIZ  /CYLINDER OVERFLOW/
9060  046066  042504  020122  053117
9061  046074  051105  046106  053517
9062  046102     000
9063  046103     111  046114  043505  EM23:   .ASCIZ  /ILLEGAL CYLINDER ADDRESS/
9064  046110  046101  041440  046131
9065  046116  047111  042504  020122
9066  046124  042101  051104  051505
9067  046132  000123
9068  046134  051104  053111  020105  EM24:   .ASCIZ  /DRIVE OFF TRACK/
9069  046142  043117  020106  051124
```

```
 9070  046150  041501  000113
 9071  046154  051104  053111  020105  EM25:   .ASCIZ  /DRIVE TIMING ERROR/
 9072  046162  044524  044515  043516
 9073  046170  042440  051122  051117
 9074  046176     000
 9075  046177     104  052101  020101  EM26:   .ASCIZ  /DATA LATE/
 9076  046204  040514  042524     000
 9077  046211     103  047117  051124  EM27:   .ASCIZ  /CONTROLLER TIMEOUT/
 9078  046216  046117  042514  020122
 9079  046224  044524  042515  052517
 9080  046232  000124
 9081  046234  050117  051105  052101  EM30:   .ASCIZ  /OPERATION INCOMPLETE/
 9082  046242  047511  020116  047111
 9083  046250  047503  050115  042514
 9084  046256  042524     000
 9085  046261     110  040505  042504  EM31:   .ASCIZ  /HEADER VRC ERROR/
 9086  046266  020122  051126  020103
 9087  046274  051105  047522  000122
 9088  046302  040504  040524  041440  EM32:   .ASCIZ  /DATA CHECK ERROR/
 9089  046310  042510  045503  042440
 9090  046316  051122  051117     000
 9091  046323     127  044522  042524  EM33:   .ASCIZ  /WRITE CHECK ERROR/
 9092  046330  041440  042510  045503
 9093  046336  042440  051122  051117
 9094  046344     000
 9095  046345     104  052101  020101  EM34:   .ASCIZ  /DATA MISCOMPARE/
 9096  046352  044515  041523  046517
 9097  046360  040520  042522     000
 9098  046365     116  020117  051104  EM35:   .ASCIZ  /NO DRIVE RESPONSE-UFE & NXD/
 9099  046372  053111  020105  042522
 9100  046400  050123  047117  042523
 9101  046406  052455  042506  023040
 9102  046414  047040  042130     000
 9103  046421     104  044522  042526  EM36:   .ASCIZ  /DRIVE ERROR WILL NOT CLEAR/
 9104  046426  042440  051122  051117
 9105  046434  053440  046111  020114
 9106  046442  047516  020124  046103
 9107  046450  040505  000122
 9108  046454  051104  053111  020105  EM37:   .ASCIZ  /DRIVE STATUS CHANGE WILL NOT CLEAR/
 9109  046462  052123  052101  051525
 9110  046470  041440  040510  043516
 9111  046476  020105  044527  046114
 9112  046504  047040  052117  041440
 9113  046512  042514  051101     000
 9114  046517     101  052124  047047  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
 9115  046524  041040  052125  047040
 9116  046532  020117  052123  052101
 9117  046540  051525  041440  040510
 9118  046546  043516  020105  051117
 9119  046554  043040  052501  052114
 9120  046562     000
 9121  046563     101  052124  047047  EM41:   .ASCIZ  /ATT'N BUT DRIVE NOT AVAILABLE/
 9122  046570  041040  052125  042040
 9123  046576  044522  042526  047040
 9124  046604  052117  040440  040526
 9125  046612  046111  041101  042514
```

```
 9126  046620     000
 9127  046621     101  052124  047047  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
 9128  046626  053440  042510  020116
 9129  046634  047516  020124  054105
 9130  046642  042520  052103  042105
 9131  046650     000
 9132  046651     105  051122  051117  EM43:   .ASCIZ  /ERROR GATHERING DRIVE STATUS/
 9133  046656  043440  052101  042510
 9134  046664  044522  043516  042040
 9135  046672  044522  042526  051440
 9136  046700  040524  052524  000123
 9137  046706  052515  052114  050111  EM52:   .ASCIZ  /MULTIPLE DRIVE SELECT/
 9138  046714  042514  042040  044522
 9139  046722  042526  051440  046105
 9140  046730  041505  000124
 9141  046734  042510  042101  051105  EM53:   .ASCIZ  /HEADER COMPARE ERROR/
 9142  046742  041440  046517  040520
 9143  046750  042522  042440  051122
 9144  046756  051117     000
 9145  046761     123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
 9146  046766  020123  044524  042515
 9147  046774  052517  000124
 9148  047000  051105  047522  020122  EM60:   .ASCIZ  /ERROR IN RECAL FOR RECOVERY/
 9149  047006  047111  051040  041505
 9150  047014  046101  043040  051117
 9151  047022  051040  041505  053117
 9152  047030  051105  000131
 9153  047034  051120  043517  040522  EM61:   .ASCIZ  /PROGRAM ABORTING FATAL ERROR IN RETRY/
 9154  047042  020115  041101  051117
 9155  047050  044524  043516  043040
 9156  047056  052101  046101  042440
 9157  047064  051122  051117  044440
 9158  047072  020116  042522  051124
 9159  047100  000131
 9160  047102  054503  044514  042116  EM62:   .ASCIZ  /CYLINDER MISCOMPARE/
 9161  047110  051105  046440  051511
 9162  047116  047503  050115  051101
 9163  047124  000105
 9164  047126  046103  040505  020122  EM63:   .ASCIZ  /CLEAR CONTROLLER DID NOT CLEAR ERROR/
 9165  047134  047503  052116  047522
 9166  047142  046114  051105  042040
 9167  047150  042111  047040  052117
 9168  047156  041440  042514  051101
 9169  047164  042440  051122  051117
 9170  047172     000
 9171  047173     116  020117  052101  EM64:   .ASCIZ  /NO ATT'N IN ATT'N SUMMARY REGISTER/
 9172  047200  023524  020116  047111
 9173  047206  040440  052124  047047
 9174  047214  051440  046525  040515
 9175  047222  054522  051040  043505
 9176  047230  051511  042524  000122
 9177  047236  047125  047523  044514  EM65:   .ASCIZ  /UNSOLICITED ATTENTION/
 9178  047244  044503  042524  020104
 9179  047252  052101  042524  052116
 9180  047260  047511  000116
 9181  047264  047125  054105  042520  EM66:   .ASCIZ  /UNEXPECTED DATA TYPE ERROR/
```

```
 9182  047272  052103  042105  042040
 9183  047300  052101  020101  054524
 9184  047306  042520  042440  051122
 9185  047314  051117     000
 9186  047317     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
 9187  047324  042040  042111  047040
 9188  047332  052117  051040  051505
 9189  047340  052105  053440  052111
 9190  047346  020110  046103  040505
 9191  047354  000122
 9192  047356  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
 9193  047364  041440  042514  051101
 9194  047372  042040  042111  047040
 9195  047400  052117  041440  042514
 9196  047406  051101  042040  044522
 9197  047414  042526  040440  052124
 9198  047422  047047     000
 9199  047425     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/
 9200  047432  040514  042524  053440
 9201  047440  042510  020116  047125
 9202  047446  047514  042101  047111
 9203  047454  020107  042510  042101
 9204  047462  051105     000
 9205  047465     103  047117  051124  EM72:   .ASCIZ  /CONTROLLER ERROR WHEN DRIVER SERVICING/
 9206  047472  046117  042514  020122
 9207  047500  051105  047522  020122
 9208  047506  044127  047105  042040
 9209  047514  044522  042526  020122
 9210  047522  042523  053122  041511
 9211  047530  047111  000107
 9212  047534  051104  053111  020105  EM73:   .ASCIZ  /DRIVE DETECTED PARITY ERROR/
 9213  047542  042504  042524  052103
 9214  047550  042105  050040  051101
 9215  047556  052111  020131  051105
 9216  047564  047522  000122
 9217  047570  047125  042504  044506  EM74:   .ASCIZ  /UNDEFINED ERROR/
 9218  047576  042516  020104  051105
 9219  047604  047522  000122
 9220  047610  040515  045522  047111  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
 9221  047616  020107  044124  051511
 9222  047624  051440  041505  047524
 9223  047632  020122  040502  000104
 9224  047640  040502  020104  040504  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
 9225  047646  040524  053040  051105
 9226  047654  043111  047047  053440
 9227  047662  052111  020110  042522
 9228  047670  042101  020056  041505
 9229  047676  020103  043117  046040
 9230  047704  051501  020124  042522
 9231  047712  051124  020131  051511
 9232  047720  000072
 9233  047722  042522  051124  020131  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
 9234  047730  052523  041503  051505
 9235  047736  043123  046125     000
 9236  047743     122  052105  054522  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
 9237  047750  052440  051516  041525
```

```
 9238  047756  042503  051523  052506
 9239  047764  000114
 9240  047766  040503  047116  052117  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
 9241  047774  043040  047111  020104
 9242  050002  020101  040526  044514
 9243  050010  020104  042510  042101
 9244  050016  051105  044440  020116
 9245  050024  051124  041501  020113
 9246  050032  052512  052123  051040
 9247  050040  040505  000104
 9248  050044  040502  020104  042523  EM102:  .ASCIZ  /BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
 9249  050052  052103  051117  042440
 9250  050060  051122  051117  047440
 9251  050066  020116  042523  052103
 9252  050074  051117  047040  052117
 9253  050102  046040  051511  042524
 9254  050110  020104  040502  000104
 9255  050116  044524  042515  026504  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
 9256  050124  052517  020124  047117
 9257  050132  051040  040505  020104
 9258  050140  042110  000122
 9259  050144  044524  042515  026504  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
 9260  050152  052517  020124  047117
 9261  050160  051440  042505  000113
 9262  050166  051104  053111  020105  EM105:  .ASCIZ  /DRIVE SIEZED BY OTHER PORT/
 9263  050174  044523  055105  042105
 9264  050202  041040  020131  052117
 9265  050210  042510  020122  047520
 9266  050216  052122     000
 9267  050221     104  052101  020101  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
 9268  050226  044515  041523  050115
 9269  050234  020122  044127  046111
 9270  050242  020105  040502  020111
 9271  050250  042523  000124
 9272  050254  047516  047040  046505  EM107:  .ASCIZ  /NO NEM ERROR WHEN REF'ING LOC 760000/
 9273  050262  042440  051122  051117
 9274  050270  053440  042510  020116
 9275  050276  042522  023506  047111
 9276  050304  020107  047514  020103
 9277  050312  033067  030060  030060
 9278  050320     000
 9279  050321     111  052116  050122  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
 9280  050326  020124  044127  047105
 9281  050334  041440  052116  046122
 9282  050342  020122  047516  020124
 9283  050350  042122  000131
 9284  050354  047516  040440  052124  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
 9285  050362  047047  047440  020116
 9286  050370  042523  045505     000  EM112:  .ASCIZ  /DRIVE'S CYLINDER INCORRECT/
 9287  050375     104  044522  042526
 9288  050402  051447  041440  046131
 9289  050410  047111  042504  020122
 9290  050416  047111  047503  051122
 9291  050424  041505  000124
 9292  050430  041101  051117  026524  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
 9293  050436  041440  047101  052047
```

```
 9294  050444  051040  040505  020104
 9295  050452  051502  000106
 9296  050456  052113  030461  043040  EM114:  .ASCIZ  /KT11 FAILURE/
 9297  050464  044501  052514  042522
 9298  050472     000
 9299  050473     115  046505  050040  EM115:  .ASCIZ  /MEM PARITY ERR/
 9300  050500  051101  052111  020131
 9301  050506  051105  000122
 9302  050512  052503  051122  047105  DH100:  .ASCIZ  /CURRENT COMMAND :/
 9303  050520  020124  047503  046515
 9304  050526  047101  020104  000072
 9305  050534  051120  053105  047511  DH105:  .ASCIZ  /PREVIOUS COMMAND :/
 9306  050542  051525  041440  046517
 9307  050550  040515  042116  035040
 9308  050556     000
 9309  050557     122  033113  030461  DH200:  .ASCIZ  /RK611 REGISTERS :/
 9310  050564  051040  043505  051511
 9311  050572  042524  051522  035040
 9312  050600     000
 9313  050601     122  046505  044501  DH500:  .ASCIZ  /REMAINING REGISTERS NOT VALID/
 9314  050606  044516  043516  051040
 9315  050614  043505  051511  042524
 9316  050622  051522  047040  052117
 9317  050630  053040  046101  042111
 9318  050636     000
 9319  050637     124  042510  043040  DH501:  .ASCIZ  /THE FOLLOWING REGISTER DATA MAY BE INCORRECT :/
 9320  050644  046117  047514  044527
 9321  050652  043516  051040  043505
 9322  050660  051511  042524  020122
 9323  050666  040504  040524  046440
 9324  050674  054501  041040  020105
 9325  050702  047111  047503  051122
 9326  050710  041505  020124  000072
 9327  050716  051105  020122  041520  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT/
 9328  050724  020040  051104  053111
 9329  050732  020105  020040  046503
 9330  050740  042116  020040  020040
 9331  050746  054503  047114  051104
 9332  050754  020040  051124  041501
 9333  050762  020113  020040  042523
 9334  050770  052103  051117  020040
 9335  050776  042127  041440  052116
 9336  051004     000
 9337  051005     110  020111  040502  DH102:  .ASCIZ  /HI BA   LO BA/
 9338  051012  020040  046040  020117
 9339  051020  040502     000
 9340  051023     120  042522  027126  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
 9341  051030  052440  044516  052502
 9342  051036  020123  040515  020120
 9343  051044  000072
 9344  051046  044510  051040  043505  DH104:  .ASCIZ  /HI REGO LO REGO/
 9345  051054  020060  047514  051040
 9346  051062  043505  000060
 9347  051066  052503  027122  052440  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
 9348  051074  044516  052502  020123
 9349  051102  040515  020120  000072
```

```
9350  051110  045522  051503  020061  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
9351  051116  020040  045522  051503
9352  051124  020062  020040  045522
9353  051132  041504  020040  020040
9354  051140  045522  040504  020040
9355  051146  020040  045522  041527
9356  051154  020122  020040  045522
9357  051162  040502  020040  020040
9358  051170  045522  051501  043117
9359  051176     000
9360  051177     122  042113  020123  DH202:  .ASCIZ  /RKDS    RKER/
9361  051204  020040  051040  042513
9362  051212  000122
9363  051214  030101  020060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
9364  051222  020040  030102  020060
9365  051230  020040  020040  030101
9366  051236  020061  020040  020040
9367  051244  030102  020061  020040
9368  051252  020040  030101  020062
9369  051260  020040  020040  030102
9370  051266  020062  020040  020040
9371  051274  030101  020063  020040
9372  051302  020040  030102  000063
9373  051310  042510  042101  051105  DH601:  .ASCIZ  /HEADER SHOULD BE:/
9374  051316  051440  047510  046125
9375  051324  020104  042502  000072
9376  051332  042510  042101  051105  DH602:  .ASCIZ  /HEADER OF SECTOR 0 THIS CYLINDER IS:/
9377  051340  047440  020106  042523
9378  051346  052103  051117  030040
9379  051354  052040  044510  020123
9380  051362  054503  044514  042116
9381  051370  051105  044440  035123
9382  051376     000
9383  051377     120  041501  020113  DH604:  .ASCIZ  /PACK ADDRESS OF ERROR(S) :/
9384  051404  042101  051104  051505
9385  051412  020123  043117  042440
9386  051420  051122  051117  051450
9387  051426  020051  000072
9388  051432  054503  047114  051104  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
9389  051440  020040  051124  041501
9390  051446  020113  020040  042523
9391  051454  052103  051117     000
9392  051461     103  046131  042116  DH6042: .ASCIZ  /CYLNDR  TRACK/
9393  051466  020122  052040  040522
9394  051474  045503     000
9395  051477     105  041503  042040  DH605:  .ASCIZ  /ECC DATA IS:/
9396  051504  052101  020101  051511
9397  051512  000072
9398  051514  047520  020123  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
9399  051522  020040  040520  020124
9400  051530  020040  020040  047503
9401  051536  051122  041505  040524
9402  051544  046102  037505     000
9403  051551     127  051117  030504  DH606:  .ASCIZ  /WORD1   WORD2   WORD3/
9404  051556  020040  053440  051117
9405  051564  031104  020040  053440
```

```
9406  051572  051117  031504     000
9407  051577     102  042101  044040  DH607:  .ASCIZ  /BAD HEADER IS :/
9408  051604  040505  042504  020122
9409  051612  051511  035040     000
9410  051617     127  020104  020043  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
9411  051624  020040  043440  047517
9412  051632  020104  020040  041040
9413  051640  042101  020040  020040
9414  051646  044040  020111  044120
9415  051654  020131  046040  020117
9416  051662  044120  020131  053040
9417  051670  052122  040440  020104
9418  051676  045440  050111  051101
9419  051704  000066
9420  051706  047507  042117  020040  DH702:  .ASCIZ  /GOOD    BAD/
9421  051714  020040  040502  000104
9422  051722  040506  046111  047111  DH703:  .ASCIZ  /FAILING DATA WORD :/
9423  051730  020107  040504  040524
9424  051736  053440  051117  020104
9425  051744  000072
9426  051746  051123  004460  051123  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
9427  051754  004461  051123  004462
9428  051762  051123  000063
9429  051766  052113  030461  051040  DH705:  .ASCIZ  /KT11 REGS :/
9430  051774  043505  020123  000072
9431  052002  042515  020115  042522  DH706:  .ASCIZ  /MEM REGS :/
9432  052010  051507  035040     000
9433  052015     120  004503  047514  DH707:  .ASCIZ  /PC     LOERAD  HIERAD  MEMSYS/
9434  052022  051105  042101  044011
9435  052030  042511  040522  004504
9436  052036  042515  051515  051531
9437  052044     000
9438  052045     120  004503  051503  DH710:  .ASCIZ  /PC     CSR AD  CSR/
9439  052052  020122  042101  041411
9440  052060  051123     000
9441  052063     103  046131  047111  DH711:  .ASCIZ  /CYLINDERS :/
9442  052070  042504  051522  035040
9443  052076     000
9444  052077     122  042113  020123  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
9445  052104  020040  051040  042513
9446  052112  020122  020040  051040
9447  052120  046513  031122  020040
9448  052126  051040  046513  031522
9449  052134     000
9450  052135     105  051122  051117  DH502:  .ASCIZ  /ERROR TRYING TO GET FAILURE INFO/
9451  052142  052040  054522  047111
9452  052150  020107  047524  043440
9453  052156  052105  043040  044501
9454  052164  052514  042522  044440
9455  052172  043116  000117
9456  052176  042523  047503  042116  DH503:  .ASCIZ  /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
9457  052204  052040  046511  047505
9458  052212  052125  047440  041503
9459  052220  051125  042522  020104
9460  052226  042507  052124  047111
9461  052234  020107  051104  053111
```

```
9462  052242  020105  052123  052101
9463  052250  051525  000
9464  052253  116  046525  042502  DH800:  .ASCIZ  /NUMBER OF RETRIES:/
9465  052260  020122  043117  051040
9466  052266  052105  044522  051505
9467  052274  000072
9468                                  .EVEN
9469  052276  001116  005430  005426  DT100:  .WORD  $ERRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
9470  052304  001162  001164  001166
9471  052312  001170
9472  052314  001256  001260          DT102:  .WORD  $REG36,$REG37
9473  052320  001174  001176  001200  DT201:  .WORD  $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
9474  052326  001202  001204  001206
9475  052334  001210
9476  052336  001212  001214          DT202:  .WORD  $REG14,$REG15
9477  052342  001216  001220  001222  DT203:  .WORD  $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
9478  052350  001224  001226  001230
9479  052356  001232  001234
9480  052362  001174  001176  001200  DT601:  .WORD  $REG5,$REG6,$REG7
9481  052370  001202  001204  001206  DT602:  .WORD  $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
9482  052376  001210  001212  001214
9483  052404  001216
9484
9485  052406  000006                  DF01:   .WORD   6               ;NUMBER OF HEADER LINES
9486  052410     000                          .BYTE   0               ;NUMBER OF COL FOR FIRST HDR
9487  052411     000                          .BYTE   0               ;ALL CHARACTERS OCTAL
9488  052412  050716                          .WORD   DH101           ;SECOND HDR LINE
9489  052414     007     000                  .BYTE   7,0             ;NUM OF COL-ALL OCTAL
9490  052416  051005                          .WORD   DH102
9491  052420     002     000                  .BYTE   2,0
9492  052422  050557                          .WORD   DH200
9493  052424     000     000                  .BYTE   0,0
9494  052426  051110                          .WORD   DH201
9495  052430     007     000                  .BYTE   7,0
9496  052432  050601                          .WORD   DH500
9497  052434     000     000                  .BYTE   0,0
9498
9499  052436  000007                  DF02:   .WORD   7
9500  052440     000     000                  .BYTE   0,0
9501  052442  050716                          .WORD   DH101
9502  052444     007     000                  .BYTE   7,0
9503  052446  051005                          .WORD   DH102
9504  052450     002     000                  .BYTE   2,0
9505  052452  050557                          .WORD   DH200
9506  052454     000     000                  .BYTE   0,0
9507  052456  051110                          .WORD   DH201
9508  052460     007     000                  .BYTE   7,0
9509  052462  051177                          .WORD   DH202
9510  052464     002     000                  .BYTE   2,0
9511  052466  051214                          .WORD   DH203
9512  052470     010     000                  .BYTE   10,0
9513
9514  052472  000010                  DF03:   .WORD   10
9515  052474     000     000                  .BYTE   0,0
9516  052476  050716                          .WORD   DH101
9517  052500     007     000                  .BYTE   7,0
```

```
 9518  052502  051005                        .WORD   DH102
 9519  052504     002      000               .BYTE   2,0
 9520  052506  050557                        .WORD   DH200
 9521  052510     000      000               .BYTE   0,0
 9522  052512  051110                        .WORD   DH201
 9523  052514     007      000               .BYTE   7,0
 9524  052516  050637                        .WORD   DH5G1          ;"THE FOLLOWING REG DATA MB INCORRECT"
 9525  052520     000      000               .BYTE   0,0
 9526  052522  051177                        .WORD   DH202
 9527  052524     002      000               .BYTE   2,0
 9528  052526  051214                        .WORD   DH203
 9529  052530     010      000               .BYTE   10,0
 9530
 9531  052532  000003                DF04:   .WORD   3
 9532  052534     000      000               .BYTE   0,0
 9533  052536  050716                        .WORD   DH101
 9534  052540     007      000               .BYTE   7,0
 9535  052542  051005                        .WORD   DH102
 9536  052544     002      000               .BYTE   2,0
 9537
 9538  052546  000007                DF05:   .WORD   7              ;OPI, HVRC
 9539  052550     000      000               .BYTE   0,0
 9540  052552  050716                        .WORD   DH101
 9541  052554     007      000               .BYTE   7,0
 9542  052556  051005                        .WORD   DH102
 9543  052560     002      000               .BYTE   2,0
 9544  052562  051310                        .WORD   DH601
 9545  052564     000      000               .BYTE   0,0
 9546  052566  051551                        .WORD   DH606
 9547  052570     003      000               .BYTE   3,0
 9548  052572  051332                        .WORD   DH602
 9549  052574     000      000               .BYTE   0,0
 9550  052576  051551                        .WORD   DH606
 9551  052600     003      000               .BYTE   3,0
 9552
 9553  052602  000007                DF07:   .WORD   7
 9554  052604     000      000               .BYTE   0,0
 9555  052606  050716                        .WORD   DH101
 9556  052610     007      000               .BYTE   7,0
 9557  052612  051005                        .WORD   DH102
 9558  052614     002      000               .BYTE   2,0
 9559  052616  051377                        .WORD   DH604
 9560  052620     000      000               .BYTE   0,0
 9561  052622  051432                        .WORD   DH6041
 9562  052624     003      000               .BYTE   3,0
 9563  052626  051477                        .WORD   DH605
 9564  052630     000      000               .BYTE   0,0
 9565  052632  051514                        .WORD   DH6051
 9566  052634     003      000               .BYTE   3,0
 9567
 9568  052636  000005                DF10:   .WORD   5
 9569  052640     000      000               .BYTE   0,0
 9570  052642  050716                        .WORD   DH101
 9571  052644     007      000               .BYTE   7,0
 9572  052646  051005                        .WORD   DH102
 9573  052650     002      000               .BYTE   2,0
```

```
9574  052652  051377                      .WORD   DH604
9575  052654     000      000             .BYTE   0,0
9576  052656  051432                      .WORD   DH6041
9577  052660     003      000             .BYTE   3,0
9578
9579  052662  000004              DF11:   .WORD   4
9580  052664     000      000             .BYTE   0,0
9581  052666  051377                      .WORD   DH604
9582  052670     000      000             .BYTE   0,0
9583  052672  051432                      .WORD   DH6041
9584  052674     003      000             .BYTE   3,0
9585  052676  051617                      .WORD   DH701
9586  052700     000      000             .BYTE   0,0
9587
9588  052702  000006              DF12:   .WORD   6
9589  052704     000      000             .BYTE   0,0
9590  052706  050716                      .WORD   DH101
9591  052710     007      000             .BYTE   7,0
9592  052712  051005                      .WORD   DH102
9593  052714     002      000             .BYTE   2,0
9594  052716  050557                      .WORD   DH200
9595  052720     000      000             .BYTE   0,0
9596  052722  051110                      .WORD   DH201
9597  052724     007      000             .BYTE   7,0
9598  052726  052077                      .WORD   DH204
9599  052730     004      000             .BYTE   4,0
9600
9601  052732  000006              DF13:   .WORD   6               ;FORMAT FOR 2ND LEVEL ERROR
9602  052734     000      000             .BYTE   0,0             ;IN HEADER COMPARE ERROR
9603  052736  050716                      .WORD   DH101           ;AND 2ND LEVEL HEADER
9604  052740     007      000             .BYTE   7,0             ;VRC ERROR
9605  052742  051005                      .WORD   DH102
9606  052744     002      000             .BYTE   2,0
9607  052746  051310                      .WORD   DH601
9608  052750     000      000             .BYTE   0,0
9609  052752  051551                      .WORD   DH606
9610  052754     003      000             .BYTE   3,0
9611  052756  052135                      .WORD   DH502
9612  052760     000      000             .BYTE   0,0
9613
9614  052762  000006              DF14:   .WORD   6               ;FORMAT FOR 2ND LEVEL ERROR
9615  052764     000      000             .BYTE   0,0             ;IN OPERATION INCOMPLETE ERROR
9616  052766  050716                      .WORD   DH101
9617  052770     007      000             .BYTE   7,0
9618  052772  051005                      .WORD   DH102
9619  052774     002      000             .BYTE   2,0
9620  052776  051310                      .WORD   DH601
9621  053000     000      000             .BYTE   0,0
9622  053002  051551                      .WORD   DH606
9623  053004     003      000             .BYTE   3,0
9624  053006  052135                      .WORD   DH502
9625  053010     000      000             .BYTE   0,0
9626
9627  053012  000011              DF15:   .WORD   11
9628  053014     000      000             .BYTE   0,0
9629  053016  050716                      .WORD   DH101
```

```
9630  053020  000007  000          .BYTE   7,0
9631  053022  051005               .WORD   DH102
9632  053024  000002  000          .BYTE   2,0
9633  053026  050557               .WORD   DH200
9634  053030  000000  000          .BYTE   0,0
9635  053032  051110               .WORD   DH201
9636  053034  000007  000          .BYTE   7,0
9637  053036  051177               .WORD   DH202
9638  053040  000002  000          .BYTE   2,0
9639  053042  052176               .WORD   DH503
9640  053044  000000  000          .BYTE   0,0
9641  053046  050637               .WORD   DH501
9642  053050  000000  000          .BYTE   0,0
9643  053052  051214               .WORD   DH203
9644  053054  000010  000          .BYTE   10,0
9645
9646  053056  000011          DF16: .WORD   11
9647  053060  000000  000          .BYTE   0,0
9648  053062  050716               .WORD   DH101
9649  053064  000007  000          .BYTE   7,0
9650  053066  051005               .WORD   DH102
9651  053070  000002  000          .BYTE   2,0
9652  053072  050557               .WORD   DH200
9653  053074  000000  000          .BYTE   0,0
9654  053076  051110               .WORD   DH201
9655  053100  000007  000          .BYTE   7,0
9656  053102  051177               .WORD   DH202
9657  053104  000002  000          .BYTE   2,0
9658  053106  052135               .WORD   DH502
9659  053110  000000  000          .BYTE   0,0
9660  053112  050637               .WORD   DH501
9661  053114  000000  000          .BYTE   0,0
9662  053116  051214               .WORD   DH203
9663  053120  000010  000          .BYTE   10,0
9664
9665  053122  000005          DF17: .WORD   5
9666  053124  000000  000          .BYTE   0,0
9667  053126  050716               .WORD   DH101
9668  053130  000007  000          .BYTE   7,0
9669  053132  051005               .WORD   DH102
9670  053134  000002  000          .BYTE   2,0
9671  053136  052063               .WORD   DH711
9672  053140  000000  000          .BYTE   0,0
9673  053142  051706               .WORD   DH702
9674  053144  000002  000          .BYTE   2,0
9675
9676  053146  000002          DF20: .WORD   2
9677  053150  000000  000          .BYTE   0,0
9678  053152  051046               .WORD   DH104
9679  053154  000002  000          .BYTE   2,0
9680
9681  053156  000002          DF21: .WORD   2
9682  053160  000000  000          .BYTE   0,0
9683  053162  051514               .WORD   DH6051
9684  053164  000003  000          .BYTE   3,0
9685
```

```
 9686  053166  000006                  DF22:   .WORD   6
 9687  053170     000      000                .BYTE   0,0
 9688  053172  050512                          .WORD   DH100
 9689  053174     000      000                .BYTE   0,0
 9690  053176  050716                          .WORD   DH101
 9691  053200     007      000                .BYTE   7,0
 9692  053202  051005                          .WORD   DH102
 9693  053204     002      000                .BYTE   2,0
 9694  053206  051066                          .WORD   DH106
 9695  053210     000      000                .BYTE   0,0
 9696  053212  051046                          .WORD   DH104
 9697  053214     002      000                .BYTE   2,0
 9698
 9699  053216  000002                  DF23:   .WORD   2
 9700  053220     000      000                .BYTE   0,0
 9701  053222  052253                          .WORD   DH800
 9702  053224     001      000                .BYTE   1,0
 9703
 9704  053226  000001                  DF24:   .WORD   1
 9705  053230     002      000                .BYTE   2,0
 9706
 9707  053232  000000                  DF25:   .WORD   0
 9708  053234     007      000                .BYTE   7,0
 9709
 9710  053236  000002                  DF26:   .WORD   2
 9711  053240     002      000                .BYTE   2,0
 9712  053242  051214                          .WORD   DH203
 9713  053244     010      000                .BYTE   10,0
 9714
 9715  053246  000005                  DF27:   .WORD   5
 9716  053250     000      000                .BYTE   0,0
 9717  053252  050716                          .WORD   DH101
 9718  053254     007      000                .BYTE   7,0
 9719  053256  051005                          .WORD   DH102
 9720  053260     002      000                .BYTE   2,0
 9721  053262  051722                          .WORD   DH703
 9722  053264     000      000                .BYTE   0,0
 9723  053266  051706                          .WORD   DH702
 9724  053270     002      000                .BYTE   2,0
 9725
 9726  053272  000005                  DF30:   .WORD   5
 9727  053274     000      000                .BYTE   0,0
 9728  053276  050716                          .WORD   DH101
 9729  053300     007      000                .BYTE   7,0
 9730  053302  051005                          .WORD   DH102
 9731  053304     002      000                .BYTE   2,0
 9732  053306  051766                          .WORD   DH705
 9733  053310     000      000                .BYTE   0,0
 9734  053312  051746                          .WORD   DH704
 9735  053314     004      000                .BYTE   4,0
 9736
 9737  053316  000005                  DF31:   .WORD   5
 9738  053320     000      000                .BYTE   0,0
 9739  053322  050716                          .WORD   DH101
 9740  053324     007      000                .BYTE   7,0
 9741  053326  051005                          .WORD   DH102
```

```
9742  053330  000002  000000                  .BYTE   2,0
9743  053332  052002                          .WORD   DH706
9744  053334  000000                          .BYTE   0,0
9745  053336  052045                          .WORD   DH710
9746  053340  000003  000000                  .BYTE   3,0
9747
9748  053342  000400                  RWBUF:  .BLKW   256.            ;READ/WRITE DATA BUFFER
9749
9750  054342  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
9751  054350  020040  020040  020040
9752  054356  025052  020052  047516
9753  054364  042524  025040  025052
9754  054372  005015  000012
9755  054375  000117  046116  020131          .ASCII  /ONLY ALL RK06 OR ALL RK07'S MUST BE UNDER TEST/<15><12><12>
9756  054402  046101  020114  045522
9757  054410  033060  047440  020122
9758  054416  046101  020114  045522
9759  054424  033460  051447  046440
9760  054432  051525  020124  042502
9761  054440  052440  042116  051105
9762  054446  052040  051505  006524
9763  054454  005012
9764  054456  046101  020114  051104          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
9765  054464  053111  051505  052040
9766  054472  020117  042502  052040
9767  054500  051505  042524  020104
9768  054506  052515  052123  044040
9769  054514  053101  020105  006472
9770  054522  005012
9771  054524  027061  042040  051505          .ASCII  /1. DESIRED PORT SELECTED/<15><12>
9772  054532  051111  042105  050040
9773  054540  051117  020124  042523
9774  054546  042514  052103  042105
9775  054554  005015
9776  054556  027062  053440  044522          .ASCII  /2. WRITE LOCK DISABLED/<15><12>
9777  054564  042524  046040  041517
9778  054572  020113  044504  040523
9779  054600  046102  042105  005015
9780  054606  051104  053111  051505          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
9781  054614  047040  052117  052040
9782  054622  020117  042502  052040
9783  054630  051505  042524  020104
9784  054636  052515  052123  044040
9785  054644  053101  020105  047502
9786  054652  044124  005015
9787  054656  047520  052122  020123          .ASCII  /PORTS DE-SELECTED./<15><12>
9788  054664  042504  051455  046105
9789  054672  041505  042524  027104
9790  054700  005015
9791  054702  044124  020105  040503          .ASCII  /THE CARTRIDGE MUST BE FORMATTED ON A KNOWN GOOD/<15><12>
9792  054710  052122  044522  043504
9793  054716  020105  052515  052123
9794  054724  041040  020105  047506
9795  054732  046522  052101  042524
9796  054740  020104  047117  040440
9797  054746  045440  047516  047127
```

```
 9798   054754   043440   047517   006504
 9799   054762      012
 9800   054763      127   046105   026514          .ASCII  /WELL-ALIGNED DRIVE, PRIOR TO TESTING./<15><12>
 9801   054770   046101   043511   042516
 9802   054776   020104   051104   053111
 9803   055004   026105   050040   044522
 9804   055012   051117   052040   020117
 9805   055020   042524   052123   047111
 9806   055026   027107   005015
 9807   055032   044124   020105   047503          .ASCII  /THE COMPLETE RUN TIME IS 8-10 MIN. PER DRIVE./<15><12>
 9808   055040   050115   042514   042524
 9809   055046   051040   047125   052040
 9810   055054   046511   020105   051511
 9811   055062   034040   030455   020060
 9812   055070   044515   027116   050040
 9813   055076   051105   042040   044522
 9814   055104   042526   006456      012
 9815   055111      124   020117   041101          .ASCIZ  /TO ABORT TESTING, TYPE CONTROL-C (↑C)./<15><12><12>
 9816   055116   051117   020124   042524
 9817   055124   052123   047111   026107
 9818   055132   052040   050131   020105
 9819   055140   047503   052116   047522
 9820   055146   026514   020103   057050
 9821   055154   024503   006456   005012
 9822   055162      000
 9823
 9824
 9825
 9826
 9827
 9828
 9829            000001                   .END
```

```
ABASE = 177440           1497#   1593    1634    3600
ABORT = 000200           3244#   3838    4939    6635    6640    6723    6748    6845    6855    6860    6865    6870    6881
                         6887    6902    6937
ACDW1 = 000000           1593
ACDW2 = 000000           1593
ACLO  = 000010           2562#   6685
ACPUOP= 000000           1593    1608
ADDW0 = 000000           1593
ADDW1 = 000000           1593
ADDW10= 000000           1593
ADDW11= 000000           1593
ADDW12= 000000           1593
ADDW13= 000000           1593
ADDW14= 000000           1593
ADDW15= 000000           1593
ADDW2 = 000000           1593
ADDW3 = 000000           1593
ADDW4 = 000000           1593
ADDW5 = 000000           1593
ADDW6 = 000000           1593
ADDW7 = 000000           1593
ADDW8 = 000000           1593
ADDW9 = 000000           1593
ADEVCT= 000000           1593    1599
ADEVM = 000377           1498#   1593    1635
ADSCOR  017644           4116    4186    4248    4535#
AENV  = 000000           1593    1604
AENVM = 000000           1593    1605
AFATAL= 000000           1593    1596
ALLSYS  013002           3594    3617#   3630    3631    3674    3704    3705    3722    5058    5130    5797
ALLSY1  013040           3627#   3711
ALLSY2  013222           3667#   3677
ALLSY3  013220           3666#   3814
ALLTRM  033336           3839    4940    6724    6868    6875#
ALNPAK  011366           3420#   4999
AMADR1= 000000           1593    1621
AMADR2= 000000           1593    1625
AMADR3= 000000           1593    1628
AMADR4= 000000           1593    1631
AMAMS1= 000000           1593    1615
AMAMS2= 000000           1593    1623
AMAMS3= 000000           1593    1626
AMAMS4= 000000           1593    1629
AMSGAD= 000000           1593    1601
AMSGLG= 000000           1593    1602
AMSGTY= 000000           1593    1595
AMTYP1= 000000           1593    1616
AMTYP2= 000000           1593    1624
AMTYP3= 000000           1593    1627
AMTYP4= 000000           1593    1630
ANOTHR  010174           3300#   3702
ANYDER= 100000           3250#   4991    6358    6925
APASS = 000000           1593    1598
APRIOR= 000000           1593
APTCSU= 000040           8265    8910#
APTENV= 000001           8258    8704    8866    8908#
```

```
APTSIZ= 000200           3559    8907#
APTSPO= 000100           8260    8868    8909#
ASKSYS  013446           3697    3708    3713    3719#   3723    3724    3726    3731    3742    5062    5134    5811
ASKTYP  010025           3281#   5482
ASWREG= 000000           1593    1606
AS2SP2  045412           6397    9001#
ATESTN= 000000           1593    1597
AUNIT = 000000           1593    1600
AUSWR = 000000           1593    1607
AVECT1= 120210           1496#   1593    1632    3599
AVECT2= 000000           1593    1633
AVSCOR  020024           4124    4193    4257    4577#
A.ABNL  003056           2843#   3842*   3845*   4869*   4943*   4948*   5802*   5984*   6001*   6010*   6194*   6209*   6611*
                         6876*   6879*   6893*   6896*   6899*   7826
A.CONT  003060           2844#   7836
A.NORM  003054           2842#   4870*   6612*   6880*   7830
BADDRV  011303           3409#   4381*   4382*   4386    4946*   4951*   4952*   4953    4965*   4966*   4967    4973*   4974*
                         4975    4996*   4997*   4998
BADSEC= 001000           3246#   5726    5911    6274    6282    6818    6930
BAD632  012036           3479#   4993
BAI   = 000020           2521#   7995
BA16  = 000400           2497#
BA17  = 001000           2498#
BDSCHD  027004           5984    6001    6015#
BDSECT  012007           3473#   6033
BDSRCK  030042           5724    5909    6264#   6817
BGNRTY  033560           6535    6890    6901    6923#
BIT0  = 000001           1428#   2494    2511    2537    2559    2707    2851    2944    3794    5646    5695    5735    5922
                         6494    8110
BIT00 = 000001           1418#   1428
BIT01 = 000002           1417#   1427
BIT02 = 000004           1416#   1426
BIT03 = 000010           1415#   1425
BIT04 = 000020           1414#   1424
BIT05 = 000040           1413#   1423
BIT06 = 000100           1412#   1422
BIT07 = 000200           1411#   1421
BIT08 = 000400           1410#   1420    5537
BIT09 = 001000           1409#   1419    8714    8840
BIT1  = 000002           1427#   2512    2539    2708    2852    2945    3238    6498
BIT10 = 002000           1408#   2499    2528    2550    2585    2599    2612    2627    2641    2720    2861    3247    6527
                         8692
BIT11 = 004000           1407#   2500    2529    2551    2570    2586    2600    2613    2628    2642    2721    3248
BIT12 = 010000           1406#   2502    2530    2552    2587    2601    2614    2615    2629    2643    2722    3249    6409
                         6437
BIT13 = 020000           1405#   2503    2531    2553    2571    2588    2602    2616    2630    2644    2723    8699
BIT14 = 040000           1404#   2504    2532    2554    2572    2589    2603    2617    2631    2645    2724    2862    8824
BIT15 = 100000           1403#   2505    2506    2533    2555    2573    2590    2650    2725    2863    3250    6530
BIT2  = 000004           1426#   2513    2540    2541    2561    2709    2853    2946    3239    6502
BIT3  = 000010           1425#   2519    2520    2542    2562    2710    2854    2947    3240    6509
BIT4  = 000020           1424#   2514    2521    2543    2563    2564    2579    2621    2635    2711    2855    2948    3241
                         6513
BIT5  = 000040           1423#   2522    2523    2544    2565    2580    2594    2607    2622    2636    2712    2856    3242
                         4741    6517    7907
BIT6  = 000100           1422#   2495    2524    2545    2566    2581    2595    2608    2623    2637    2713    2858    3243
                         4746    4779    5762    5949
```

```
BIT7   = 000200   1421# 2496  2525  2546  2567  2568  2582  2596  2609  2624  2638  2715  3244
                  3597  3791  3804  4955  5095
BIT8   = 000400   1420# 2497  2526  2547  2548  2569  2583  2597  2610  2625  2639  2716  2859
                  3245  6521
BIT9   = 001000   1419# 2498  2527  2549  2584  2598  2611  2626  2640  2717  2860  3246  5084
                  6524  8804
BKSLSH  012137    3491# 5384  5392
BLDEXH  034444    6764  6794  7077*
BPTVEC= 000004    1435#
BSE    = 000200   2546# 6015  6809  7260  7426  7662
BSERR  = 000002   3238# 6197  6811  6825  6923
BSFACT  004162    2954# 5983  6031  6265
BSSOFT  003162    2953# 5995  6064  6270  6272
BUFFO   005212    2959# 3637  3641  3643  3648  3659  3661  3663  3709  3712  3714  3727  3729
                  3732  3734  3775  4349* 4350* 4352* 4354  4357  4362  4367* 4384* 4389  4393*
                  4822  4826  5051  5063  5123  5135  5387  5401* 5406* 5407  5411* 5489  5493
                  5503
BUFPRT  005422    2962#
B.BA16= 000001    2511# 7989
B.BA17= 000002    2512# 7989
B.CDT  = 000004   2513# 5505  7643  7989
B.CFMT= 000020    2514# 4988  5086  5460  5973  7033  7643  7989  8073
CART    011740    3465# 5470
CCLR   = 100000   2506# 7596
CDT    = 002000   2499# 3827  4928  7402  7952  8009  8060  8083
CERR   = 100000   2505# 6773  6803  7298  7357  7416  7598  7649  7964  8020  8108
CFMT   = 010000   2502# 7056  7089  7402  7952  8009  8060  8083
CKMEM1  026112    5886#
CKMEM2  026124    5890#
CKMO    026132    5889  5892#
CKSCOR  023220    5272# 6200
CLEAR  = 000105   2465#
CLR    = 000040   2522#
CLRPRM  021270    4874  4892#
CMDTO  = 000100   2713# 6658  6988  7159
CMPBUF  025044    5682#
CMSTRT  012236    3515  3520# 6914
CNTRDY  011537    3440# 5816
CNTRLC  012102    3485# 4745  5355
CNTRLG  012126    3489# 4759  5373
CNTRLR  012114    3487# 4755
CNTRLU  012121    3488# 5367
CNTRLZ  012107    3486# 4751  5361
COE    = 001000   2549# 6730  7260  7426  7662
COMAVG  007430    3211# 4383  4635* 4647* 4648* 4675
COMMA   012141    3492# 3687
COMSTR  005176    2957# 6329  6338  6942
CONCLR= 000176    2483# 8091
CONERR  031000    2844  6490#
CR     = 000015   1343# 8304  8314
CRLF   = 000200   1344# 8275  8314
CRTSER  024056    5094  5459#
CR2LF   012133    3490# 6396  6408  6432
CTLOUT  025564    4338  5789# 6310
CTO    = 004000   2500# 6693  7658
CYLNDR  005434    2967#
```

```
C.INIT  040222          6315    7869#
C.RTRN  042074          7973    8001    8029    8036    8039    8046    8050    8067    8089    8094    8104    8113    8126
                        8130    8135#
C.SPEC  041114          7909    8005#
DCEFLG  003144          2931#   4725*   6192*
DCK     = 100000        2555#   4721    6015    6827    7260    7426    7662
DCKERR= 000020          3241#   4724    5275    6132    6157    6829    6923
DCKHDL  020556          4716#   6194    6876    6896
DCLO    = 000020        2564#
DDISP   = 177570        1350#   1527    3547
DDT     = 000400        2569#
DERCNT  003130          2919#   6757    6787    6836    6847
DESL    = 000010        2519#
DFLTDR  013774          3607    3789#
DFSTRT  012176          1454    3510#   5057    5129
DF01    052406          1658    1664    1670    1676    1718    1826    2096    2132    9485#
DF02    052436          1682    1700    1706    1712    1724    1730    1736    1742    1748    1754    1760    1766    1772
                        1778    1784    1790    1844    1850    1856    1964    1970    1976    1982    1988    1994    2000
                        2006    2012    2018    2060    2078    2102    9499#
DF03    052472          1688    1694    2066    2072    9514#
DF04    052532          1820    9531#
DF05    052546          1796    1802    9538#
DF07    052602          1808    9553#
DF10    052636          1814    9568#
DF11    052662          2114    9579#
DF12    052702          1862    1868    1874    1880    1886    1892    1898    1904    9588#
DF13    052732          1910    1922    9601#
DF14    052762          1916    9614#
DF15    053012          1928    9627#
DF16    053056          1934    9646#
DF17    053122          1952    9665#
DF20    053146          2120    9676#
DF21    053156          2030    9681#
DF22    053166          2126    9686#
DF23    053216          2036    2042    2048    9699#
DF24    053226          2054    9704#
DF25    053232          1958    5684#   5714*   5896*   5914*   9707#
DF26    053236          2108    9710#
DF27    053246          2084    9715#
DF30    053272          2138    9726#
DF31    053316          2144    9737#
DH100   050512          1656    1662    1668    1674    1680    1686    1692    1698    1704    1710    1716    1722    1728
                        1734    1740    1746    1752    1758    1764    1770    1776    1782    1788    1794    1800    1806
                        1812    1818    1824    1842    1848    1854    1860    1866    1872    1878    1884    1890    1896
                        1902    1908    1914    1920    1926    1932    1950    1962    1968    1974    1980    1986    1992
                        1998    2004    2010    2016    2058    2064    2070    2076    2082    2094    2100    2130    2136
                        2142    9302#   9688
DH101   050716          6413    9327#   9488    9501    9516    9533    9540    9555    9570    9590    9603    9616    9629
                        9648    9667    9690    9717    9728    9739
DH102   051005          6424    9337#   9490    9503    9518    9535    9542    9557    9572    9592    9605    9618    9631.
                        9650    9669    9692    9719    9730    9741
DH103   051023          2118    9340#
DH104   051046          9344#   9678    9696
DH105   050534          6411    9305#
DH106   051066          9347#   9694
DH200   050557          9309#   9492    9505    9520    9594    9633    9652
```

```
DH201   051110          9350#  9494   9507   9522   9596   9635   9654
DH202   051177          9360#  9509   9528   9637   9656
DH203   051214          9363#  9511   9526   9643   9662   9712
DH204   052077          9444#  9598
DH500   050601          9313#  9496
DH501   050637          9319#  9524   9641   9660
DH502   052135          9450#  9611   9624   9658
DH503   052176          9456#  9639
DH601   051310          9373#  9544   9607   9620
DH602   051332          9376#  9548
DH604   051377          9383#  9559   9574   9581
DH6041  051432  6034    9388#  9561   9576   9583
DH6042  051461  2052    9392#
DH605   051477  2028    9395#  9563
DH6051  051514          9398#  9565   9683
DH606   051551          9403#  9546   9550   9609   9622
DH607   051577          9407#
DH701   051617          5683#  5713#  5895#  5913#  9410#  9585
DH702   051706  2106    9420#  9673   9723
DH703   051722          9422#  9721
DH704   051746          9426#  9734
DH705   051766          9429#  9732
DH706   052002          9431#  9743
DH707   052015          9433#
DH710   052045          9438#  9745
DH711   052063          9441#  9671
DH800   052253          9464#  9701
DI      =  040000       2504#  7446   8118
DIF100  003147          2934#  4082#  4118#  4159#  4231#  4254#  4472   4477#
DISPAK  010404          3324#  5110#  5111#  5114
DISPLA  001142          1527#  3547#  3555#  8691#  8851#
DISPRE  000174          1451#  3555
DLT     =  100000       2533#  6750   7254   7384   7420   7660
DLTFLG  003146          2933#
DMCART  022472  4003    4400   5108#
DMD     =  000040       2580#
DONE    003132          2921#  6311#  6318   6357#  6491#  6605#  6957#  6960#
DRA     =  000001       2559#  7249   8032
DRDY    =  000200       2568#
DRIDON  010642          3352#  5116
DRIRDY  010623          3349#  5044
DRIV    011731          3463#  5428
DRIVE   005430          2965#  3819#  3822   4696   4877   4878   4922   4951   4965   4973   4996   5067#  5068#
                        9469
DRNAFG  003141          2928#  6312#  6504
DRNAVL= 010000          3249#
DROPDR  011641          3453#
DROT    =  000040       2565#  6740
DROVFL  010245          3307#  3673
DRPAR   =  000010       2542#  6680
DRPDRV= 002000          2720#  7894
DRVBUF  011230          3398#  4345#  4347#  4348
DRVCAL  030146          3844   3921   3924   4086   4177   4180   4475   4478   4508   4514   4517   4876   4945
                        4980   4982   4990   5017   5075   5427   5469   5805   5846   5986   6003   6196   6308#
                        6894   6941   6951   6987   7024   7026
DRVDSC= 000040          2712#  6857   7334   7500   7562
```

```
DRVERS  003137          2926#   3567*   6381*
DRVHRD= 000020          2711#   6852    7341    7536    7667
DRVLST  005512          2985#   3620    3625    3671    3735    3739    3806    4016    4262    4282    4335    4414    4632
                        5037    5160
DRVMSK= 000007          2518#   7243    7284    7435    7880
DRVNO   011725          3462#
DRVPDT= 000004          2709#   7442    7510
DRVPOS= 000002          2708#   7442    7469    7931
DRVPTR  005554          2989#   3744*   3809*   4004*   4005    4016*   4329    4341    4380    4401*   4402    4414*   5034
                        5067    5109
DRVSEQ  011265          3406#
DRVSER  023736          5093    5424#
DRVSZD= 010000          2722#   6671    7272    8034
DRVTST  013734          3700    3774    3779#
DRVUSE= 000001          2707#   7894
DRY     = 000200        2567#
DR.CLR= 000005          2490#   7322    7485    7527    7551    7665
DR.SEL= 000001          2489#   7392    7471    7786    7793    7800    7807    8011
DROXDP  011610          3448#
DSC     = 040000        2572#
DSWR    = 177570        1349#   1526    3546
DTBAII= 100000          2725#   4095    4240    6140    7894    7993
DTE     = 010000        2552#   6015    6745    7662
DTYE    = 000040        2544#   6708    7662
DT100   052276          1657    1663    1669    1675    1681    1687    1693    1699    1705    1711    1717    1723    1729
                        1735    1741    1747    1753    1759    1765    1771    1777    1783    1789    1795    1801    1807
                        1813    1819    1825    1843    1849    1855    1861    1867    1873    1879    1885    1891    1897
                        1903    1909    1915    1921    1927    1933    1951    1963    1969    1975    1981    1987    1993
                        1999    2005    2011    2017    2059    2065    2071    2077    2083    2095    2101    2125    2131
                        2137    2143    9469#
DT102   052314          9472#
DT201   052320          9473#
DT202   052336          2107    9476#
DT203   052342          9477#
DT601   052362          2029    2035    2041    2047    2053    2113    9480#
DT602   052370          1957    9481#
DZR6Q   007572          3252#   3571
ECCNC   = 000040        3242#   6832
ECCW    = 020000        2588#
ECH     = 000100        2545#   6830    7260    7426    7662
ECOBAD  021014          4793#
EMTVEC= 000030          1438#   3531*   3532*
EM1     045417          1655    9002#
EM10    045632          1697    9029#
EM100   047743          2045    9236#
EM101   047766          2051    9240#
EM102   050044          2057    9248#
EM103   050116          2063    9255#
EM104   050144          2069    9259#
EM105   050166          2075    9262#
EM106   050221          2081    9267#
EM107   050254          2087    9272#
EM11    045641          1703    9031#
EM110   050321          2093    9279#
EM111   050354          2099    9284#
EM112   050375          2105    9287#
```

EM113	050430	2129	9292#	
EM114	050456	2135	9296#	
EM115	050473	2141	9299#	
EM12	045654	1709	9033#	
EM13	045675	1715	9036#	
EM14	045717	1721	9040#	
EM15	045745	1727	9044#	
EM16	045766	1733	9047#	
EM17	046003	1739	9050#	
EM2	045443	1661	9006#	
EM20	046024	1745	9053#	
EM21	046041	1751	9056#	
EM22	046061	1757	9059#	
EM23	046103	1763	9063#	
EM24	046134	1769	9068#	
EM25	046154	1775	9071#	
EM26	046177	1781	9075#	
EM27	046211	1787	9077#	
EM3	045467	1667	2147	9010#
EM30	046234	1793	1913	9081#
EM31	046261	1799	1919	9085#
EM32	046302	1805	9088#	
EM33	046323	1811	9091#	
EM34	046345	1817	2117	9095#
EM35	046365	1823	9098#	
EM36	046421	1829	9103#	
EM37	046454	1835	9108#	
EM4	045512	1673	9014#	
EM40	046517	1841	9114#	
EM41	046563	1847	9121#	
EM42	046621	1853	9127#	
EM43	046651	1859	9132#	
EM5	045533	1679	9017#	
EM52	046706	1901	9137#	
EM53	046734	1907	9141#	
EM56	046761	1925	1931	9145#
EM6	045552	1685	9020#	
EM60	047000	1937	9148#	
EM61	047034	1943	9153#	
EM62	047102	1949	9160#	
EM63	047126	1865	1961	9164#
EM64	047173	1871	1967	9171#
EM65	047236	1877	1973	9177#
EM66	047264	1883	1979	9181#
EM67	047317	1889	1985	9186#
EM7	045576	1691	9024#	
EM70	047356	1895	1991	9192#
EM71	047425	1997	9199#	
EM72	047465	2003	9205#	
EM73	047534	2009	9212#	
EM74	047570	2015	9217#	
EM75	047610	2021	9220#	
EM76	047640	2027	9224#	
EM77	047722	2033	2039	9233#
ENDTST	011170	3392#	4413	
EQUALS	012164	3501#		

```
ERRCNT  003135          2924#  3568*  5014*  6195*  6360*  6362   6365   6367*  6492*  6606*  6777*  6822*  6912*
                        6932   6935   6952*
ERRCOM  005426          2964#  6543*  6544*  9469
ERRFRE  030272          2842   4870   6357#  6880
ERRHDL  031504          2843   3845   4691   4723   4869   4948   5802   6010   6017   6209   6604#  6879   6899
ERRLMT  003136          2925#  6932
ERRVEC= 000004          1431#  3544   3545*  3556*  3564*  3565*  8829   8830*  8832*  8835*
ER2ENT  031524          6608#  6670   6783   6808   6904
E.CCLR= 000001          2851#  7601
E.CERR= 001000          2860#  7675
E.CLAT= 000020          2855#  7327   7491   7532   7556
E.CMTO= 040000          2862#
E.CONT  003062          2845#  6494   6498   6502   6509   6513   6517   6521   6524   6527   6530   6534*  7226*
                        7264*  7327*  7337*  7408*  7430*  7453*  7459*  7491*  7532*  7556*  7601*  7653*  7675*
                        7814*  8048*  8102*  8110*  8123*
E.DLT = 000400          2859#  7408
E.DPAR= 002000          2861#  7337   7814
E.ILLD= 000100          2858#  8102
E.MDS = 100000          2863#  7226   7653   8048
E.NOAT= 000002          2852#  7453
E.SCLR= 000040          2856#  8123
E.UATT= 000004          2853#  7459
E.UDAT= 000010          2854#  7264   7430
E.UNLD= 020000          2723#  7345   7465   7540   7550
FACTBS  011762          3469#  6032
FINADR  024542          5589#  5719   5904
FINCYL  005470          2977#  5619*  5721   5739   5745   5906   5926   5932
FINMEM  027202          6076#  6127
FINSEC  005473          2979#  5607*  5723   5743   5747   5908   5930   5934
FINTRK  005472          2978#  5617*  5722   5741   5746   5907   5928   5933
FMTE  = 000020          2543#  6712   7662
FORMAT  003134          2923#  4097   4242   4879   5082*  5086*  5461   5599   5978   5998   6103   7034
GETPRM  021026          3748   3753   3762   4808#  4848
GETTYP  024176          3592   5481#  5484   5485   5486   5491
GNS   = ****** U        1450   8987   8988   8989   8990   8991   8994   8995   8996   8997
GO    = 000001          2494#  3825   3827   4925   4928
GTPKAD  034310          5278   6838   6849   7044#
GTPRMS  027540          6169#  6206
GTSWRG  021136          3589   4844#  5374   5822
GTO     027552          6168   6172#
HCRC  = 000400          2547#
HDR.AD  003100          2885#  7378   7382*  8072*
HDR.CT  003102          2886#  7386*  8075*  8078*
HIGOCT  005420          2961#
HLPOVL  003152          2938#  3570*
HLTPRG  033542          5817   5994   6910#  6938
HLTRQD  011567          3445#  5815
HOLD1   024406          5076   5497*  5507*  5515#  7027
HOLD2   024410          5079   5498*  5508*  5516#  7030
HOLD3   024412          4102   4117   4190   4253   5499*  5509*  5517#
HOLD4   024414          4238   5500*  5510*  5518#
HOLD5   024416          4474   4476   5501*  5511*  5519#
HT    = 000011          1341#  8273   8314
HVRC  = 000400          2548#  4717   6015   6784   7260   7426   7662
HVRCER= 000004          3239#  4719   5275   6197   6786   6923   7049
H.HEAD  003105          2889#
```

```
IDAE   = 002000   2550#  6735
IE     = 000100   2495#  3821  4921  4959  5041  5113  6769  6799  7606  7670  7958  7999  8065
                  8132
ILC    = 000001   2537#  6696  7662
ILF    = 000004   2540#  6704
INCRSC  027266    6088   6102#  6135
INITSS  021170    3820   3840  3874  3897  3945  4000  4045  4079  4151  4228  4325  4869#  4920
                  4941   5040  5065  5073  5087  5112  5137  5425  5459  5972
INTBLD  022642    3947   3967  4114  4237  5150#
INTCHR  005442    2970#  4742* 4761  4778* 4793  5098* 5791  5795  5800* 5809  5813  5820
INTMSK  003111    2894#  7301  7325  7346  7371  7457  7489  7530  7541  7554  7825* 7829* 7833*
                  7882*  7968
INTR   = 000300   2485#
IOTVEC= 000020    1436#  3529* 3530*
IR     = 000100   2524#
I.AERR  036604    7477   7483  7501  7510#
I.ATTN  036134    7237   7281  7414#
I.CCLR  037170    7315   7365  7596#  7668  8028  8093
I.CSTS  037622    7308   7743#
I.CSTI  037644    3835   4936  7312  7364  7747#  8017
I.DRV   003112    2898#  7882  7883
I.ERR   035356    7309   7313#
I.ERRA  035336    7308#  7376
I.ERRC  035344    7246   7299  7311#  7444
I.HDAL  035602    7292   7355#
I.INTR  034722    3579   7216#
I.ISRL  003104    2888#  7230  7233* 7236* 7607* 7904* 8058* 8131*
I.ISSU  037252    7323   7393  7472  7486  7528  7552  7638#  7787  7794  7801  7808
I.IOO   035202    7234   7251  7278#
I.RTRN  037152    7228   7266  7276  7304  7316  7321  7324  7330  7339  7348  7351  7366  7368
                  7394   7397  7406  7410  7432  7448  7455  7461  7473  7487  7494  7506  7526
                  7529   7534  7543  7546  7553  7558  7567#
I.STAT  037726    7320   7525  7784#  8038
I.STOR  037474    7648   7698#
I.UNLD  037052    7467   7550#
KBDHDL  020634    3576   4734#
LASTWC  005474    2980#  5590  5718* 5903*
LDMEM1  026042    5858#
LDMEM2  026054    5862#
LDMO    026062    5861   5864#
LEV2ER= 000400    3245#  6608  6663  6760  6766  6790  6796  6900  6958
LF     = 000012   1342#  8308  8314
LODBUF  024720    5637#
LODSEC  024502    3901   4164  5561#
LOGDRV  003154    2940#  3903  4089  4112  4161  5039* 5234  5240
LOWOCT  005416    2960#
LSTCYL  024404    4984   5465  5496* 5506* 5514#  5975
LSTTRK= 000002    3237#  4985  5466
MA      005500    2982#
MCLK   = 000400   2583#
MDFLAG  003126    2917#  3510* 3513* 3517* 3593  3698  4017  4415  5055  5059  5089  5127  5131
                  5793
MDS    = 001000   2527#  3831  4932  7224  7651  8026
MEMABT  003151    2936#
MEMVEC= 000114    2951#
MERD   = 001000   2584#
```

```
MESMSK= 000017           2577#
MEWD  = 002000           2585#
MIDXFR  027332           6126#   6201
MIND  = 000200           2582#
MNTPAK  010340           3318#   5035*   5036*   5042
MSP   = 000100           2581#
MTCART  022110           3876    4048    5033#
M.CADD= 017760           2653#
M.CDIF= 017760           2652#
M.DPID  042676           4603    4668    8357#
M.DPIM  042636           8328#
M.DPO1  042646           8331#   8340
M.DPO2  042666           8335    8339#
M.DP40  042734           8368#   8389
M.DP41  042770           8373    8380#
M.DP42  043000           8379    8384#
M.DP44  043032           8391    8396#
M.DP50  043044           8366    8400#
M.DRV = 000007           2649#
M.HEAD= 007000           2656#
M.ID  = 000003           2651#
M.PAR = 100000           2650#
M.SECT= 000760           2655#
M.SER = 077770           2654#
NCSCR0  007112           3163#   4232    4234    4260    4277    4555    4590    4637
NCSCR1  007152           3166#
NCSCR2  007212           3169#
NED   = 010000           2530#   4689    6642    6652    7239    7317    7660    8030
NEDHDL  020500           3842    4689#   4943
NEM   = 004000           2529#   6637    7254    7420    7660
NEWMSG  011254           3404#   4812
NEWON   005476           2981#   3843*   3846    4701*   4944*   4949
NGOFST  005452           2974#
NOCHK = 000400           2716#   7894    7956    7997    8024    8042    8063    8086    8095    8129
NODRLS  010652           3354#   3741
NODRTS  011506           3435#
NODSC = 004000           2721#   6862    7482
NOFALS  012027           3476#   6041
NORTRY  003150           2935#   4873*   6928
NOSCR0  006612           3141#   4107    4109    4333    4551    4585    4636    4653
NOSCR1  006652           3144#
NOSCR2  006712           3147#
NOTMSG  054342           3574    9750#
NOTYPE  003153           2939#   3572    3575*
NSCOR0  007432           3216#   4168*   4466    4468    4490    4496    4538    4558    5302
NSCOR1  007452           3219#   4169*   4540
NSCOR2  007472           3222#   4170*   4542
NSSCR0  007412           3186#   4153*   4261    4539*   4580
NSSCR1  007414           3189#   4154*   4541*
NSSCR2  007416           3192#   4155*   4543*
NTREDY  011333           3414#   4968
NXDRIV  011314           3411#   4954
NXF   = 000004           2541#
OAVERG  011132           3386#   4385
OCTBIN  042116           4827    8160#
OFFSET= 000115           2469#   4507    4516    7939
```

```
OFINUS  005436          2968#
OFSCNT  003157          2943#  4482*  4501   4509*  5313
OFSDIR  003156          2942#  4479*  4491   4498   4523*  4594*  4600   4609*  5303
OFST   = 000004          2561#
ONSUBS  010154          3297#  3693*  3694
OPCOMP  003131          2920#
OPCONT  003140          2927#
OPI    = 020000          2553#  6015   6754   7260   7426   7662
OPIERR= 000010          3240#  6756   6923
OR     = 000200          2525#
OREQSW= 000020          2948#
OVRAVG  007426          3208#  4384   4634*  4645*  4646*  4662
O.WAIT  003064          2865#  7160   7164*  7274*  7278   7294*  7360*  7373*  7388*  7932*  7992*  8054*  8071*
PACK   = 000103          2464#  4979
PARM0   002640          2754#  2909   4872   5015   6162   6171   6875   6891   6895*  6949   7001   7035
PARM1   002724          2786#  4878*  6610   6891*  6895   6985   7019
PASSNO  003155          2941#  3514*  3518*  3779   3781   4047*  5091   5818*
PAT    = 000020          2579#
PATRN   005446          2972#  5639   5688   5764
PATOO   005566          2992#  5641   5656   5690   5705
PBLKT   003122          2909#  7157   7244   7270   7438   7463   7881*
PBSVAL= 001000          2717#  7819
PCA    = 004000          2586#
PCD    = 010000          2587#
PCSCR0  007252          3174#  4280   4297   4556   4591
PCSCR1  007312          3177#
PCSCR2  007352          3180#
PGE    = 002000          2528#  6700   7660
PIP    = 020000          2571#
PIRQ   = 177772          1348#
PIRQVE= 000240          1442#
PLOFST  005450          2973#
PMA     005504          2983#  5658   5707   5756   5758   5865   5893   5943   5945   6145*  6146*  6147*  6148
                        6149   6166*  6167*
POSCR0  006752          3152#  4552   4586
POSCR1  007012          3155#
POSCR2  007052          3158#
PREPKB  020774          3875   3898   3946   4001   4046   4080   4152   4229   4326   4777#  5823
PRMBUF  012152          3499#
PRMPSP  012173          3503#
PROMPT  012170          3502#  4359
PRVCMD  005162          2955#  3583   6328   6577
PR0    = 000000          1365#  3582   5018   6950   7023
PR1    = 000040          1366#
PR2    = 000100          1367#
PR3    = 000140          1368#
PR4    = 000200          1369#  3577
PR5    = 000240          1370#  2841
PR6    = 000300          1371#
PR7    = 000340          1372#  3520   8784   8785
PS     = 177776          1345#  1346   3520*  3582*  5018*  7146   7152*  7166*  7875   7876*  8135*
PSCOR0  007512          3227#  4171*  4500   4544   4559   5305
PSCOR1  007532          3230#  4172*  4546
PSCOR2  007552          3233#  4173*  4548
PSSCR0  007420          3197#  4156*  4281   4545*  4581
PSSCR1  007422          3200#  4157*  4547*
```

```
PSSCR2  007424  3203# 4158* 4549*
PSW   = 177776  1346# 6950* 7023*
PWRMSG  044540  8787  8791#
PWRVEC= 000024  1437# 3535* 3536* 8775* 8783* 8784*
P.ASOF= 000032  2738# 6564  7488* 7489  7522* 7753* 7941* 7942
P.A00 = 000040  2741# 4963  4971  6567  6991  7479* 7480  7496* 7497  7810* 8018*
P.A01 = 000044  2743# 6569  7789*
P.A10 = 000050  2745# 6571  7796*
P.A11 = 000054  2747# 5430  6573  7803*
P.BAHI= 000007  2700# 6151* 6553
P.BALO= 000010  2701# 3917* 4094* 4163* 4239* 4986* 5467* 5842* 5983* 5995* 6148* 6556  7975
                8072
P.BAR = 000024  2735# 6563  7519* 7749*
P.B00 = 000042  2742# 6568  7474* 7475  7811* 8019*
P.B01 = 000046  2744# 6570  7790*
P.B10 = 000052  2746# 6572  7013  7797*
P.B11 = 000056  2748# 6574  7395* 7804*
P.CMND= 000001  2694# 3833* 3841* 3920* 3923* 4084* 4088* 4176* 4179* 4471* 4507* 4510* 4513*
                4516* 4875* 4934* 4942* 4979* 4981* 4983* 5016* 5074* 5426* 5464* 5804* 5845*
                5974* 6544  6892* 6940* 6986* 7022* 7025* 7290  7907  7911  7923  7933  7939
                7945  7948  7954  7979  7986  8005  8052  8069  8091  8100
P.CS1 = 000016  2732# 3834* 4895  4935* 6557  6675  6693  7056  7089  7311* 7363* 7401* 7402*
                7404* 7405  7514* 7516* 7743* 7746* 7897  7951* 7952* 7954* 7958* 7959  7983*
                7986* 7988* 7989* 7999* 8000  8008* 8009* 8011* 8012  8014* 8015  8020  8059*
                8060* 8062* 8065* 8066  8082* 8083* 8085* 8088  8107* 8108  8117* 8118
P.CS1H= 000007  2699# 4879* 4880* 4988* 4989* 5460* 5973* 7033* 7034* 7401  7642  7951  7988
                8008  8059  8073  8082
P.CS2 = 000020  2733# 4689  6558  6632  6637  6642  6644  6650  6652  6700  6750  6843  7317
                7517* 7747* 7905* 7930  7995* 7996  8007  8026  8030  8056* 8057  8081
P.CYLN= 000002  2695# 3908* 3913* 3951* 3954* 3957* 3960* 3970* 3973* 4085* 4090* 4102* 4113*
                4117* 4175* 4178* 4246* 4249* 4474* 4476* 4984* 5251* 5465* 5618  5975* 6077
                6136* 6546  7021* 7399  7935  7977  8079
P.DCYL= 000030  2737# 6081  6128  6559  6812  7044  7078  7521* 7755*
P.DRVN= 000000  2693# 4877* 7287  7464  7640  7879
P.DS  = 000036  2740# 6565  6685  6689  6740  7524* 7751* 8032
P.DTS = 000026  2736# 6083  6085  6129  6130  6560  6813  6815  7012  7047  7048  7079  7085
                7520* 7750*
P.EPAT= 000062  2750# 6840  7313* 7899
P.EPOS= 000060  2749# 6839  7314*
P.ER  = 000034  2739# 4717  4721  6015  6566  6631  7523* 7752*
P.OFST= 000006  2698# 4506* 4515* 7941
P.PRST= 000014  2703# 4095* 4240* 6140  6658  6671  6852  6857  6862  6867  6988  7159* 7272*
                7306* 7334* 7341  7345* 7442* 7465  7469* 7482* 7500* 7510* 7536  7540* 7550*
                7562* 7667* 7819* 7894* 7931* 7955* 7956  7981  7987* 7993  7997  8024  8034*
                8042  8063  8086  8095  8129
P.SECT= 000004  2696# 3910* 3915* 4166* 4187* 4188  5244* 5463* 5595  5980* 5989* 5990  5996*
                6006* 6007  6079  6138* 6549  7020* 7400  7936  7978  8080
P.TRCK= 000005  2697# 3926* 3927  4167* 4183* 4184  4485  4985* 5245* 5247* 5248  5466* 5609
                5976* 6078  6137* 6547
P.WC  = 000012  2702# 3918* 4096* 4099* 4165* 4241* 4244* 4987* 5468* 5843* 5982* 6139* 6164
                6202  6551  7976
P.WCR = 000022  2734# 6561  7518* 7748*
P1SET   012212  3511  3514#
P1STRT  012204  1456  3513#
P2STRT  012222  1458  3517#
QNEWSW= 000000  2453
```

```
Q.INIT= 040000          2724#
RAVERG  011151          3389#   4394
RCDASW= 000010          2947#
RCLREQ= 004000          3248#   6678    6683    6728    6733    6738    6743    6780    6889
RDALHD= 000164          2482#   7290    8069
RDCHR = 104406          5350    8994#
RDCHRS  023424          3629    3703    3721    4813    5045    5117    5344#   5483
RDDATA= 000121          2471#   4510    4983    5464    5974
RDGATE= 100000          2590#
RDHDO   034134          6765    6795    7010#
RDHEAD= 000125          2473#   5074    7025    7404    8085
RDSTAT= 000141          2481#   3841    4942    5426    6986    8005
RDY   = 000200          2496#   5077    5080    6885    7028    7031    7962    8015
RECAL = 000113          2468#   4981    5804    6892    7948
RECODE  005424          2963#   3838*   4716*   4719*   4724*   4939*   4991    5013*   5275    5726    5911    6132    6193*
                        6197    6274*   6282*   6358    6368*   6607*   6608    6635*   6640*   6661    6663    6678*   6683*
                        6723*   6728*   6733*   6738*   6743*   6748*   6756*   6760    6766    6780*   6786*   6790    6796
                        6811*   6818    6825*   6829*   6832*   6845*   6846*   6855*   6860*   6865*   6870*   6881    6887*
                        6889    6900    6902*   6923    6925*   6930    6937*   6958*   6990*   7049
REDBSF  026540          5088    5971#
REDOFS  017300          4115    4182    4247    4462#
REISSU  003142          2929#   6506*
RELEAS= 000140          2480#   8052
REPLPK  011417          3425#
REPSUP  031224          3836    4937    5687    5897    5992    6541#   6613
RESLTS  010717          3361#   4330*   4331*   4332
RESREG= 104410          4525    4566    4618    4678    5193    5219    5254    5316    5449    5571    5621    5668    5725
                        5774    5847    5873    5910    5960    6067    6092    6152    6176    6275    6345    6477    6592
                        6953    7002    7036    7097    8432    8484    8996#
RESVEC= 000010          1432#
RETANL  033736          6611    6893    6957#
RETNML  033754          4704    4726    6019    6612    6960#
RKASOF= 000016          2449#   6979    7300    7370    7450    7705    7753    7942*   7967
RKBA  = 000004          2444#   6978    7701    7749    7975*
RKBADR  007660          3261#   3747
RKBAS   003046          2839#   3596*   3600*   3746    3749*   4871    6768    6798    6884    6939    6971    7153    7222
                        7886
RKCS1 = 000000          2442#   4959*   5041*   5076*   5077    5079*   5080    5113*   6769*   6773    6799*   6803    6973
                        7027*   7028    7030*   7031    7245    7297    7355    7405*   7414    7596*   7597    7606*   7645*
                        7646    7670*   7698    7959*   7961    8000*   8012*   8014    8044*   8066*   8088*   8097*   8107
                        8117    8132*
RKCS2 = 000010          2446#   3822*   3831    4922*   4932    6974    7223    7383    7641*   7699    7747    7930*   7996*
                        8007*   8057*   8081*   8106*
RKDA  = 000006          2445#   6976    7400*   7702    7750    7936*   7978*   8080*
RKDB  = 000024          2452#   5083    5084    6770    6771    6772    6800    6801    6802    7379    7380    7381
RKDC  = 000020          2450#   6975
RKDCYL= 000020          2451#   7399*   7706    7755    7935*   7977*   8079*
RKDS  = 000012          2447#   6885    6980    7248    7703    7751
RKECPS= 000030          2457#   7314    7710
RKECPT= 000032          2459#   7313    7711
RKER  = 000014          2448#   6981    7257    7423    7704    7752
RKMR1 = 000026          2453#   7391*   7470*   7707    7784*   7791*   7798*   7805*   7906*
RKMR2 = 000034          2454#   7708    8018
RKMR3 = 000036          2455#   7709    8019
RKPAT = 000032          2458#
RKPOS = 000030          2456#
```

```
RKPRI   003052          2841#   3580    3605*   3756    3772*   7152    7876
RKPRTY  007727          3269#   3761
RKVADR  007704          3265#   3752
RKVEC   003050          2840#   3578    3601*   3602*   3751    3754*
RKWC  = 000002          2443#   6977    7700    7748    7976*
RLS   = 000010          2520#   8056
ROTBLD  023022          3965    4120    4252    5204#
RWBUF   053342          3917    4093*   4094    4163    4239    4986    4994    5467    5472    5562    5566    5643    5692
                        5836    5842    6166    9748#
R.ABNL  040154          7165    7275    7350    7367    7545    7565    7825#   8035
R.CONT  040200          7227    7265    7329    7338    7409    7431    7454    7460    7493    7533    7557    7602    7654
                        7677    7815    7833#   8049    8103    8112    8125
R.NORM  040166          7303    7396    7505    7829#   7972    8045    8133
SAVPRS  005454          2975#   6163    6170
SAVREG= 104407          4462    4535    4577    4630    5150    5204    5233    5272    5424    5561    5589    5637    5682
                        5720    5834    5858    5862    5886    5890    5905    6029    6076    6126    6161    6169    6264
                        6327    6380    6490    6542    6604    6970    7011    7077    8417    8458    8995#
SAVWRD  003160          2952#
SCLR  = 000040          2523#   8106
SCNDRV  021316          4920#   5069
SCOPER= 104411          8997#
SCOPE1  044562          8802#   8997
SCRACH  005444          2971#   3680*   3683*   3688    3828*   3829*   4929*   4930*   5385*   5387*   5388    5398*   5399*
                        5400    5686*   5728    5734*   5737    5894*   5915    5921*   5924
SCRLST  005556          2990#   3653    3661*   3667    3683    3795*   3798*   3799*   3817    3852*
SECLST  006572          3131#   5244    5291
SEEK  = 000117          2470#   4084    4176    4471    7923    7933
SELDRV= 000101          2463#   3833    4934    8062
SELF    011235          3399#   4343
SERNM   011747          3467#   5429    5471
SETTUP  022042          3873    3896    3944    3999    4044    4078    4150    4227    4324    5011#
SKI   = 000002          2539#   6725
SOFTBS  011775          3471#   6065
SPACE1  012150          3497#   4356
SPACE2  012147          3496#   4361    6046    6052    6419    6423    6426    6429    6449    6454    6469
SPACE3  012146          3495#
SPACE4  012145          3494#
SPACE6  012143          3493#
SPAR  = 020000          2503#   6675    7658
SPDLSS= 000020          2563#   6689
SRHTBS  027700          6225#   6266
SRPASS  010307          3313#   3779*   3780    4019*   4020
SRTSPL= 000111          2467#   7945
STACK = 001100          1336#   3527    5011*   5012    5054    5061    5126    5133    5803    8786
STALL   024420          5535#   6309
STALLS  005432          2966#   3588*   5544
STDLST  006552          3121#   5152
STKLMT= 177774          1347#
STRCMD  030220          6313    6327#
STR204  010450          3330#   4012*   4013
STR220  010525          3338#   4022*   4023    4409*   4410
SUBCLR= 000177          2484#   4513    4875    5016    6940    7022    8100
SUBSYS  005510          2984#   3606*   3734*   3736    3807    4008    4405
SVAL  = 100000          2573#
SVPRMS  027500          4100    4511    6161#   6207
SWR     001140          1526#   3525    3546*   3548    3554*   3561*   4844    5095    5537    5735    5762    5922    5949
```

```
                        6382    6409    6437    8692    8699    8711    8714    8804    8824    8840
SWREG   000176          1452#   3554    4844    4846    4849*
SWRMSG  011245          3402#   4847
SW0     =  000001       1400#
SW00    =  000001       1390#   1400
SW01    =  000002       1389#   1399
SW02    =  000004       1388#   1398
SW03    =  000010       1387#   1397
SW04    =  000020       1386#   1396
SW05    =  000040       1385#   1395
SW06    =  000100       1384#   1394
SW07    =  000200       1383#   1393
SW08    =  000400       1382#   1392
SW09    =  001000       1381#   1391
SW1     =  000002       1399#
SW10    =  002000       1380#
SW11    =  004000       1379#
SW12    =  010000       1378#
SW13    =  020000       1377#   6382
SW14    =  040000       1376#
SW15    =  100000       1375#
SW2     =  000004       1398#
SW3     =  000010       1397#
SW4     =  000020       1396#
SW5     =  000040       1395#
SW6     =  000100       1394#
SW7     =  000200       1393#
SW8     =  000400       1392#
SW9     =  001000       1391#
SYSDRV  010103          3289#   3627*   3628
S.ACLO= 000100          2608#
S.BRHM= 000100          2623#
S.BRKE= 040000          2645#
S.CART= 000400          2625#
S.DCLO= 010000          2615#
S.DIB = 002000          2641#
S.DOOR= 000200          2624#
S.DRA = 000040          2594#
S.DROT= 020000          2616#
S.DRY = 000200          2596#   4963
S.DSC = 040000          2603#   7332    7480    7497    7560
S.FLT = 000200          2609#   7475
S.FORM= 001000          2598#
S.FWD = 002000          2627#
S.HDFL= 000200          2638#
S.HDHM= 000040          2622#
S.ICYL= 000040          2607#
S.ILF = 000400          2610#
S.LIMD= 020000          2644#
S.LOAD= 010000          2629#
S.MHD = 000400          2639#
S.NMOV= 010000          2643#
S.OFF = 002000          2599#
S.PAR = 001000          2611#   7335    7812
S.PIP = 020000          2602#   7343    7538
S.PLO = 004000          2642#
```

```
S.REV  = 004000          2628#
S.RTZ  = 020000          2630#
S.SECT= 000020           2635#
S.SKI  = 002000          2612#
S.SPIN= 010000           2601#
S.SPLS= 010000           2614#
S.SPOK= 001000           2626#
S.TYPE= 000400           2597#
S.UNLD= 040000           2631#
S.UNS  = 040000          2617#
S.VV   = 000100          2595#
S.WCLK= 000040           2636#
S.WGAT= 000100           2637#
S.WLE  = 004000          2613#
S.WRL  = 004000          2600#   4971
S.XDOK= 000020           2621#
S.XERR= 001000           2640#
TAB     011243           3401#   4365
TABLEA  005566           2998#   3908    3910
TABLEB  005766           3038#   5563
TABLEC  006026           3059#   3951    3954    3970
TABLED  006112           3080#   5153    5168    5182    5185    5205    5217    5237    5289
TABLEG  006512           3100#   4092    5236
TABL7A  005666           3018#   3913    3915
TABL7C  006060           3069#   3957    3960    3973
TBITVE= 000014           1433#
TKVEC  = 000060          1440#   3576*   3577*
TOPROC  033764           6493    6660    6969#
TPVEC  = 000064          1441#
TRACK   005440           2969#
TRAPVE= 000034           1439#   3533*   3534*   8785*
TRKAVG  020262           4379    4630#
TRKBUF  011224           3397#   4336*   4339
TRNSFR  027570           4101    4512    6190#
TRTVEC= 000014           1434#
TSTDRN  011704           3459#
TSTING  003127           2918#   3581*   3899*   4002*   4081*   4327*   4399*   5789    5801*
TSTMSG  045400           6403    8999#
TSTTYP  003143           2930#   3566*   4106*   4125*   4174*   4181*   4194*   4230*   4258*   4464    4483    4536    4553
                         4578    4588    5273    5279    5806*   6775    6820    6877    6897
TST1    014346           3783    3871#   4011
TST10   016016           4225#
TST11   016434           4322#
TST2    014376           3894#
TST3    014570           3942#
TST4    014762           3997#
TST5    015146           3785    4021    4042#   4408
TST6    015204           4076#
TST7    015510           4148#
TWOTOS= 002000           3247#   6661    6990
TYPBSF  027030           5097    6029#
TYPDS  = 104405          8991#
TYPE   = 104401          3571    3574    3617    3628    3637    3638    3673    3679    3687    3688    3694    3702    3714
                         3715    3720    3729    3730    3741    3747    3752    3761    3773    3775    3776    3780    4013
                         4020    4023    4332    4339    4343    4348    4356    4359    4361    4365    4370    4373    4386
                         4387    4392    4397    4410    4413    4745    4751    4755    4759    4761    4762    4793    4794
```

```
                        4812    4822    4823    4847    4953    4954    4967    4968    4975    4976    4993    4998    4999
                        5042    5043    5044    5051    5052    5114    5115    5116    5123    5124    5355    5361    5367
                        5373    5384    5388    5392    5400    5405    5407    5408    5410    5428    5429    5448    5470
                        5471    5475    5481    5482    5489    5490    5815    5816    6032    6033    6034    6035    6041
                        6042    6046    6052    6056    6065    6396    6397    6398    6402    6403    6408    6411    6412
                        6413    6414    6419    6422    6423    6424    6425    6426    6429    6432    6433    6435    6449
                        6454    6458    6464    6469    6470    6472    8278    8525    8590    8665    8694    8702    8787
                        8987#
TYPERR   030352         6380#   8701
TYPFMT   003133         2922#   3823    3906    3949    3968    4880    4923    4989    5495*   5505*
TYPOC =  104402         6045    6051    6055    6418    6428    6431    6451    8988#
TYPON =  104404         8990#
TYPOS =  104403         4809    6405    8989#
TYPRWN   010602         3346#   5043    5115
TYPSYS   007752         3273#   3720
T.ASOF   003024         2826#   7300*   7301    7325    7370*   7371    7450*   7451    7457    7488    7522    7530    7554
                        7705*   7967*   7968
T.BA     003016         2823#   7519    7701*
T.CS1    003010         2818#   7245*   7297*   7298    7311    7322*   7355*   7357    7363    7392*   7414*   7416    7446
                        7471*   7485*   7515*   7516    7527*   7551*   7597*   7598    7638    7642*   7643*   7645    7649
                        7658    7698*   7745*   7746    7786*   7793*   7800*   7807*   7961*   7962    7964
T.CS2    003012         2820#   7223*   7224    7239    7242    7254    7283    7383*   7384    7420    7434    7517    7639*
                        7640*   7641    7651    7660    7699*
T.DA     003020         2824#   7520    7702*
T.DB     003044         2835#
T.DC     003022         2825#   7521    7706*
T.DS     003030         2829#   7248*   7249    7524    7703*
T.ER     003026         2828#   7257*   7260    7423*   7426    7523    7662    7704*
T.MR1    003032         2830#   7707*
T.MR2    003034         2831#   7332    7343    7479    7496    7538    7560    7708*   7789    7796    7803    7810
T.MR3    003036         2832#   7335    7395    7474    7709*   7790    7797    7804    7811    7812
T.PAT    003042         2834#   7711*
T.POS    003040         2833#   7710*
T.WCR    003014         2822#   7518    7700*
UEXATT=  000010         2710#   6867    7306
UFE    = 000400         2526#   6644    6650    7239    7317    7660    8030
UNLOAD=  000107         2466#   7911
UNS    = 040000         2554#   6720
UPE    = 020000         2531#   6632    7254    7420    7660
VFHDSW=  000002         2945#
VV     = 000100         2566#
WCDASW=  000004         2946#
WCE    = 040000         2532#   6843    7254    7420    7660
WCEFLG   003145         2932#   5985*   5987    6002*   6004    6018*
WCERR  = 000100         3243#   6846    6923
WDSXFR   005466         2976#   5638    5685    5717    5864    5892    5902    6091*   6131    6164*   6165*
WHDSW  = 000001         2944#
WILTST   010130         3293#   3679
WLE    = 004000         2551#   6716    7260    7426
WORDSP   012156         3500#
WRDATA=  000123         2472#   3920    4088    4179    5845    7983
WRHEAD=  000127         2474#
WRL    = 004000         2570#
WRTBLK   023106         3952    3958    3961    3974    5233#
WRTCHK=  000131         2475#   3923    7979
WRTGAT=  040000         2589#
```

```
WRTLOK  011347          3417#   4976
WRTSEC  025770          5246    5834#
W.DRV   003124          2915#   7154#   7269#   7295#   7347#   7362#   7375#   7389#   7437#   7504#   7513#   7542#   7564#
                        7835#   7884#   7947#   7950#   7971#   8023#   8041#
W.MILI  003070          2868#   7149    8115
W.MIN   003076          2884#   7947
W.MTIM  003066          2867#   7147#   7149#
W.SEC   003072          2881#   7884
W.TIME  003110          2890#   7150    7156#   7268#   7296#   7346#   7361#   7374#   7390#   7436#   7503#   7512#   7541#
                        7563#   7834#   7883#   7970#   8022#   8040#
W.WCK = 000200          2715#   7894    7955    7981    7987
W.WTCH  034572          6317    7142#   7960    7966    8013
W.8SEC  003074          2883#   7347    7542    7950
$APTHD  001000          1483    1489#
$ASTAT= ****** U        8888    8903
$ATYC   045002          8859    8861#
$ATY1   044756          8857#
$ATY3   044764          8263    8858#
$ATY4   044774          8707    8860#
$AUTOB  001134          1523#
$BASE   001404          1634#   3596
$BDADR  001122          1518#
$BDDAT  001126          1520#
$BELL   001320          1585#   8694    8726
$CHARC  042632          8280#   8290#   8297    8306#   8311#
$CKSWR= ****** U        8994
$CMTAG  001100          1506#   3522    3523    3531    3537    3538
$CM1  = 000040          1538#   1539#   1540#   1541#   1542#   1543#   1544#   1545#   1546#   1547#   1548#   1549#   1550#
                        1551#   1552#   1553#   1554#   1555#   1556#   1557#   1558#   1559#   1560#   1561#   1562#   1563#
                        1564#   1565#   1566#   1567#   1568#   1569#   1570#
$CM2  = 000100          1538#   1539#   1540#   1541#   1542#   1543#   1544#   1545#   1546#   1547#   1548#   1549#   1550#
                        1551#   1552#   1553#   1554#   1555#   1556#   1557#   1558#   1559#   1560#   1561#   1562#   1563#
                        1564#   1565#   1566#   1567#   1568#   1569#   1570#
$CM3  = 000040          1536#   1538
$CM4  = 000016          1570#   1571#   1572#   1573#   1574#   1575#   1576#   1577#   1578#   1579#   1580#   1581#   1582#
                        1583#   1584#
$CNTLG  044423          8764#
$CNTLU  044416          8763#
$CPUOP  001356          1608#
$CRLF   001325          1587#   3617    3773    4370    4373    4392    4397    4762    5405    5410    5448    5475    5481
                        6035    6042    6056    6402    6412    6414    6422    6425    6435    6458    6464    6472    8279
                        8314    8702    8726
$DBLK   044072          8631    8665    8673#
$DB2D   043174          4363    4390    8458#
$DB2O   043054          5473    8417#
$DECVL  043354          8460    8506#
$DEVCT  001340          1599#
$DEVM   001406          1635#   3790
$DOAGN  017274          4437    4442    4448#
$DTBL   044062          8634    8669#
$ENDAD  017264          1469    4444#   8721
$ENDCT  017250          3537    4439#
$ENV    001350          1604#   8258    8704    8866    8890
$ENVM   001351          1605#   3559    3597    3791    3804    4955    8260    8265    8868
$EOP    017220          4430#
$EOPCT  017242          3537*   4436#   4440
```

```
$ERFLG  001103          1509#   8689*   8726    8802    8816    8838    8844*   8854
$ERMAX  001115          1515#   3539*   8850*   8854
$ERROR  044102          3531    8688#
$ERRPC  001116          1516#   8696*   8697*   8698    8726    9469
$ERRTB  001410          1652#   6393
$ERTTL  001112          1513#   8695*   8726
$ESCAP  001316          1584#   3538*   8717    8719    8726    8849*
$ETABL  001350          1603#
$ETEND  001410          1495    1636#
$FATAL  001332          1596#   8894*
$FFLG   045222          8857*   8860*   8888    8897*   8905#
$FILLC  001156          1534#   8283    8314
$FILLS  001155          1533#   8314
$GDADR  001120          1517#
$GDDAT  001124          1519#
$GET42  017254          4441#
$GTSWR= ****** U        8993
$HD   = 000002          1317
$HIBTS  001000          1490#
$HINUM  042350          3590*   8215    8223    8228*   8233#
$HIOCT  042250          4830    8185*   8198#
$ICNT   001104          1510#
$INTAG  001135          1524#
$ITEMB  001114          1514#   6388    8698*   8706    8726
$LF     001326          1588#   8314    8726
$LFLG   045221          8898*   8904#
$LONUM  042352          3591*   5540    8214    8221    8227*   8234#
$LPADR  001106          1511#   3540*   8789    8842*   8847*   8852    8854
$LPERR  001110          1512#   3541*   8716    8806    8842    8848*   8854
$MADR1  001362          1621#
$MADR2  001366          1625#
$MADR3  001372          1628#
$MADR4  001376          1631#
$MAIL   001330          1491    1495    1594#   3558    3872    3895    3943    3998    4043    4077    4149    4226    4323
                        8258    8704    8846
$MAMS1  001360          1615#
$MAMS2  001364          1623#
$MAMS3  001370          1626#
$MAMS4  001374          1629#
$MBADR  001002          1491#
$MFLG   045220          8858*   8864    8899*   8903#
$MNEW   044441          8767#
$MSGAD  001344          1601#   8874*   8877
$MSGLG  001346          1602#   8879*
$MSGTY  001330          1595#   8872    8880*   8892    8896*
$MSWR   044430          8765#
$MTYP1  001361          1616#
$MTYP2  001365          1624#
$MTYP3  001371          1627#
$MTYP4  001375          1630#
$NULL   001154          1532#   8285    8314
$NWTST= 000001          3859#   3861    3882#   3884    3933#   3935    3978#   3980    4029#   4031    4052#   4054    4129#
                        4131    4198#   4200    4302#   4304
$OCNT   043652          8562*   8591*   8604#
$OCTVL  043156          5431    8419    8444#
$OMODE  043654          8557*   8561*   8566    8569*   8580*   8606#
```

```
$OVER   044742            8825    8843    8851#
$PASS   001336            1598#   3558*   4433*   4434*   4450
$PASTM  001006            1493#
$PWRCT  044536            8780*   8781*   8790#
$PWRDN  044452            3535    8775#   8783
$PWRUP  044464            8775    8780#
$QUES   001324            1586#   3638    3715    3730    3776    4794    4823    5052    5124    5408    5490    8314    8726
$RAND   042252            5539    8210#
$RDCHR  044276            8742#   8994
$RDDEC= ****** U          8995
$RDLIN= ****** U          8995
$RDOCT= ****** U          8995
$RDSZ = 000000            8763#
$REGAD  001160            1536#
$REG0   001162            1538#   6545    9469
$REG1   001164            1539#   9469
$REG10  001202            1546#   5716*   5717*   5718    5749*   5752    5901*   5902*   5903    5936*   5939    6770*   6800*
                          6839*   9473    9481
$REG11  001204            1547#   5750*   5937*   6771*   6801*   6840*   9473    9481
$REG12  001206            1548#   5751*   5938*   6772*   6802*   6833*   6835*   9473    9481
$REG13  001210            1549#   5753*   5755*   5757*   5758*   5940*   5942*   5944*   5945*   9473    9481
$REG14  001212            1550#   5752*   5754*   5756*   5939*   5941*   5943*   9476    9481
$REG15  001214            1551#   5759*   5760*   5946*   5947*   9476    9481
$REG16  001216            1552#   9477    9481
$REG17  001220            1553#   9477
$REG2   001166            1540#   9469
$REG20  001222            1554#   9477
$REG21  001224            1555#   9477
$REG22  001226            1556#   9477
$REG23  001230            1557#   9477
$REG24  001232            1558#   9477
$REG25  001234            1559#   9477
$REG26  001236            1560#   6416    6576
$REG27  001240            1561#
$REG3   001170            1541#   9469
$REG30  001242            1562#
$REG31  001244            1563#
$REG32  001246            1564#
$REG33  001250            1565#
$REG34  001252            1566#
$REG35  001254            1567#
$REG36  001256            1568#   6555*   9472
$REG37  001260            1569#   6556*   9472
$REG4   001172            1542#
$REG5   001174            1543#   5283    5731*   5739    5745*   5918*   5926    5932*   6364*   6365*   6552    6934*   6935*
                          6972    7044*   7066*   7078*   7092    7095    9473    9480
$REG6   001176            1544#   5306    5309    5732*   5741    5746*   5919*   5928    5933*   7045*   7048*   7061    7063*
                          7065*   7085*   7086*   7087*   7088*   7091*   7093    7094    9473    9480
$REG7   001200            1545#   5291    5733*   5743    5747*   5920*   5930    5934*   7046*   7047*   7051    7054*   7058*
                          7060*   7092*   7094*   7096*   9473    9480
$RESRE  045262            8944#   8996
$R2A  = ****** U          8997
$SAVRE  045224            8928#   8995
$SCOPE  044606            3529    8823#
$SETUP= 000037            3506#   3528    3529    3531    3533    3535    3537    3538    3540    4432    8689    8714    8721
                          8731    8769    8824
```

```
$STUP  = 177777          3506#
$SUPRS   043370          4364    4391    5447    5474    8517#
$SVLAD   044706          8833    8839    8845#
$SVPC  = 000230          1467#   1472
$SWR   = 163000          1317#   1319#   1324    1325    1326    1327    1328    1329    1330    1584    1585    3538    3540
                         3541    3872    3895    3943    3998    4043    4077    4149    4226    4323    4427    4433    4443
                         4449    4450    8680    8681    8662    8683    8684    8692    8699    8711    8714    8726    8817
                         8818    8819    8820    8824    8836    8838    8839    8840    8845    8848    8851    8854
$SWREG   001352          1606#   3561
$SWRMK=  000000          8820
$TESTN   001334          1597#   3872*   3895*   3943*   3998*   4043*   4077*   4149*   4226*   4323*   8846*
$TKB     001146          1529#   4735    4777*   8729    8746    8752
$TKS     001144          1528#   4746*   4779*   8729    8744    8750
$TMP0    001262          1570#
$TMP1    001264          1571#   6229*   6230*   6238
$TMP10   001302          1578#   6242*   6243*   6244
$TMP11   001304          1579#   4263*   4265*   4269    4283*   4285*   4289    4353*   4368*   4383*   4395*   4580*   4585*
                         4590*   4599    4611*   4612*   4662*   4667    4674*   4675*   5208*   5214
$TMP12   001306          1580#   4269*   4270*   4271    4273*   4274    4289*   4290*   4291    4293*   4294    4333*   4337
                         4375*   4583*   4593*   4616*   4663*   4676*   5209*   5215
$TMP13   001310          1581#   4351*   4366*   4367    4581*   4586*   4591*   4602    4614*   4615*   5210*   5216
$TMP14   001312          1582#
$TMP15   001314          1583#
$TMP2    001266          1572#   6225*   6253
$TMP3    001270          1573#   6226*   6254
$TMP4    001272          1574#   6227*   6255
$TMP5    001274          1575#
$TMP6    001276          1576#
$TMP7    001300          1577#
$TN    = 000012          1306#   1317    3859    3872#   3882    3895#   3933    3943#   3978    3998#   4029    4043#   4052
                         4077#   4129    4149#   4198    4226#   4302    4323#
$TNPWR   043304          8465    8466    8486#
$TPB     001152          1531#   8303*   8314
$TPFLG   001157          1535#   8252    8314
$TPS     001150          1530#   8301    8314
$TRAP    045320          3533    8964#
$TRAP2   045342          8975#   8986
$TRP   = 000012          8979#   8988#   8989#   8990#   8991#   8992#   8994    8995#   8996#   8997#   8998#
$TRPAD   045354          8969    8986#
$TSTM    001004          1492#
$TSTNM   001102          1508#   3569*   3618*   3719*   3784*   4010*   4407*   4432*   5807*   6404    8691    8726    8816
                         8845*   8846    8851    8854
$TYPBN=  ****** U        8992
$TYPDS   043656          8619#   8991
$TYPE    042354          8252#   8885    8979    8987
$TYPEC   042566          8282    8289    8296    8301#   8302
$TYPEX   042634          8307    8309    8312#
$TYPOC   043454          8560#   8988
$TYPON   043470          8559    8562#   8990
$TYPOS   043430          8555#   8989
$UNIT    001342          1600#
$UNITM   001010          1494#
$USWR    001354          1607#
$VECT1   001400          1632#   3595
$VECT2   001402          1633#
$XTSTR   044616          8827#
```

```
$$GET4= 000000             4443#
$.$.SM= ****** U           2867    7171    7406    7597    7608    7646    7960    8001    8013    8067    8089    8107    8134
                           8143
$OFILL  043653             8556*   8560*   8570    8605#
$40CAT= ****** U           8701    8824
.       = 055163           1446#   1450#   1455#   1457#   1467    1468#   1470#   1472#   1473#   1479    1480#   1482#   1484#
                           1505#   1589    2953#   2954#   2959#   2975#   2982#   2983#   2985#   2990#   3080#   3081#   3082#
                           3083#   3084#   3085#   3086#   3087#   3088#   3089#   3090#   3091#   3092#   3093#   3094#   3095#
                           3141#   3144#   3147#   3152#   3155#   3158#   3163#   3166#   3169#   3174#   3177#   3180#   3216#
                           3219#   3222#   3227#   3230#   3233#   3526    3540    3541    4450    8314    8386    8444#   8506#
                           8673#   8726    8729    8769    8777    8794#   8854    8906#   9748#
.$ASTA= ****** U           8858    8861
.$X     = 001000           1479#   1484
..HIGH  000226             1461#
..LOW   000224             1460#
```

```
CKEXIT  2167#  6310
COMMEN  1443#
ENDCOM  1443#
ERROR   1337#  3837  4938  4957  5730  5748  5761  5917  5935  5948  5993  6366  6496  6500  6507
        6511   6515  6519  6523  6526  6529  6532  6533  6634  6639  6646  6648  6654  6656  6665
        6667   6669  6673  6677  6682  6687  6691  6695  6698  6702  6706  6710  6714  6718  6722
        6727   6732  6737  6742  6747  6752  6762  6779  6782  6792  6805  6807  6824  6841  6850
        6854   6859  6864  6869  6903  6905  6936
ESCAPE  1443#
GETPRI  1443#
GETSWR  1443#
MSG     3859#  3861  3882#  3884  3933#  3935  3978#  3980  4029#  4031  4052#  4054  4129#  4131  4198#
        4200   4302# 4304
MULT    1443#
NEWTST  1443#  3859  3882  3933  3978  4029  4052  4129  4198  4302
NEXTAD  2163#
NXTAD   2159#
PARMBK     1#  2751  2783
PARMDF     1#  2657
POP     1443#  8229  8660  8900  8901  8949
PUSH    1443#  8210  8619  8861  8863  8884  8929
P.DRVR     1#  2815
REGDEF     1#  2439
REPORT  1443#
R.DRIV     1#  7100
R.QDRV     1#
SCOPE   1338#  3871  3894  3942  3997  4042  4076  4148  4225  4322  4431
SETPRI  1443#
SETTRA  8979#  8988  8989  8990  8991  8994  8995  8996  8997
SETUP   1443#  3521
SKIP    1443#
SLASH   1443#
SPACE   1443#
STARS   1295   1302  1443#  1465  1476  1478  1485  1501  1589  1592  3611  3615  3859  3870  3882
        3893   3933  3941  3978  3996  4029  4041  4052  4075  4129  4147  4198  4224  4302  4321
        4425   4453  4461  4530  4534  4571  4576  4623  4629  4683  4688  4710  4715  4730  4732
        4767   4775  4784  4791  4802  4807  4837  4843  4859  4867  4885  4890  4905  4918  5006
        5010   5025  5032  5103  5107  5142  5149  5198  5203  5224  5232  5259  5271  5321  5342
        5419   5423  5454  5458  5527  5533  5557  5560  5579  5588  5626  5635  5673  5681  5779
        5788   5828  5833  5852  5857  5878  5885  5967  5970  6023  6028  6072  6075  6097  6101
        6119   6125  6157  6160  6181  6189  6214  6224  6257  6263  6291  6307  6324  6326  6350
        6356   6370  6379  6479  6489  6536  6594  6602  6964  6968  7007  7009  7041  7043  7071
        7076   7109  7140  7174  7214  7577  7594  7612  7636  7683  7696  7715  7741  7760  7782
        7840   7867  8145  8158  8201  8237  8317  8327  8345  8356  8408  8447  8509  8532  8609
        8676   8728  8734  8798  8801  8813  8856  8913  8958
SWRSU   1292#  1443# 3542#
S.DRVE     1#  2591
TEST    2155#  3859  3882  3933  3978  4029  4052  4129  4198  4302
TRMTRP  8979#
TYPBIN  1443#
TYPDEC  1443#
TYPNAM  1443#
TYPNUM  1443#
TYPOCS  1443#
TYPOCT  1443#
TYPTXT  1443#
```

```
$DECBN       1#
$OCTBN       1#    8143
$$CMRE    1499#    1538    1539    1540    1541    1542    1543    1544    1545    1546    1547    1548    1549    1550    1551
          1552     1553    1554    1555    1556    1557    1558    1559    1560    1561    1562    1563    1564    1565    1566
          1567     1568    1569
$$CMTM    1499#    1570    1571    1572    1573    1574    1575    1576    1577    1578    1579    1580    1581    1582    1583
$$ESCA    1443#
$$NEWT    1443#    3859    3882    3933    3978    4029    4052    4129    4198    4302
$$SET     8979#    8988    8989    8990    8991    8994    8995    8996    8997
$$SETM    3558#
$$SKIP    1443#
.EQUAT    1292#    1333
.HEADE    1292#    1307
.SETUP    1292#    3506
.SWRHI    1292#    1320
.SWRLO    1292#    1331#
.$ACTI    1292#    1463
.$APTB    1590#
.$APTH    1292#    1474
.$APTY    1292#    8854
.$CATC    1292#    1444
.$CMTA    1292#    1499
.$DB2D    1292#    8445
.$DB2O    1292#    8406
.$EOP     1292#    4423
.$ERRO    1292#    8674
.$POWE    1292#
.$RAND    1292#    8199
.$READ    1292#    8726
.$SAVE    1292#    8911
.$SCOP    1292#    8811
.$SIZE    1292#
.$SUPR    1292#    8507
.$TRAP    1292#    8956
.$TYPD    1292#    8607
.$TYPE    1292#    8235
.$TYPO    1292#    8530

. ABS.   055163     000

 ERRORS DETECTED:  0

 RM03:CZR6QB,RM03:CZR6QB.SEQ/SOL/CRF/NL:TOC/DOC=RM03:DRIV10.P11/EQ:QNEWSW,RM03:CZR6QB.P11
 RUN-TIME: 36 28 2 SECONDS
 RUN-TIME RATIO: 617/67=9.1
 CORE USED:  47K  (93 PAGES)

 DOCUMENT PAGES:  215
```