RK611/RK06

SUBSYSTEM VERIFY PART 2

MD-11-DZR6N-C

EP-DZR6N-C-DL-B

FICHE 1 OF 2

DEC 1976

digital

MADE IN USA

RK611/RK06

SUBSYSTEM VERIFY PART 2

MD-11-DZR6N-C

EP-DZR6N-C-DL-B

FICHE 2 OF 2

DEC 1976

digital

MADE IN USA

.REM 3

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZR6N-C-D

PRODUCT NAME: RK611/RK06 SUBSYSTEM VERIFICATION:PART 2

DATE: DECEMBER 1976

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: DAVE HOFFMAN

TABLE OF CONTENTS

1.0 ABSTRACT

2.0 HARDWARE REQUIREMENTS
2.1 REQUIREMENTS FOR SUBSTSTEM TESTS
2.2 REQUIREMENTS FOR HEAD ALIGNMENT AID

3.0 PRELIMINARY PROGRAM REQUIREMENTS

4.0 GENERAL PROGRAM CONSIDERATIONS
4.1 SYSMAC
4.2 XXDP
4.2.1 CHAIN MODE
4.2.2 DUMP MODE
4.3 ACT/APT
4.3.1 AUTOMATIC MODE
4.3.2 DUMP MODE
4.3.3 APT ETABLE DEFINITIONS
4.4 DUAL-ACCESS
4.5 MEMORY MANAGEMENT
4.6 MEMORY PARITY CHECK
4.7 BAD SECTORS
4.8 EXECUTION TIME

5.0 PROGRAM LOADING

6.0 STARTING PROCEDURE
6.1 STARTING ADDRESSES
6.2 SWITCH REGISTER OPTIONS USED

7.0 OPERATOR ACTION

8.0 PROGRAM ACTION
8.1 DESCRIPTION OF OPERATING PARAMETERS
8.2 SELECTION OF OPERATING PARAMETERS
8.2.1 DRIVE SELECTION
8.2.2 TEST SELECTION
8.2.2.1 LIST TESTS, (L)
8.2.2.2 CHANGE TESTS, (C)
8.2.2.3 INPUT PARAMETERS AND RUN TESTS, (I)
8.2.2.4 CONTROL Z (↑Z) FUNCTION
8.2.2.5 CONTROL C (↑C) FUNCTION
8.2.3 PARAMETER LIST ALTERATION
8.2.3.1 TYPE LIST, (T)
8.2.3.2 OPEN LIST, (O)
8.2.3.3 SET INDIVIDUAL PARAMETER, (S)
8.2.3.4 RUN TESTS, (R)

	8.2.3.5	CONTROL Z (^Z) FUNCTION
	8.2.3.6	CONTROL C (^C) FUNCTION

TABLE OF CONTENTS (CONT'D)

8.2.4 SPECIAL PARAMETER SPECIFICATIONS
8.2.4.1 PT - DATA PATTERN SELECT WORD
8.2.4.2 CS - CONTROL SWITCH WORD
8.3 DATA PATTERNS

9.0 DESCRIPTION OF TESTS
9.1 TEST 1 - OFFSET-TO-FAILURE MEASUREMENT
9.2 NPR/MAIN MEMORY TESTS
9.2.1 TEST 2 - NPR/MEMORY WORD ADDRESSING TEST
9.2.2 TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST
9.2.3 TEST 4 - NPR/MEMORY DATA PATTERN TEST
9.3 TEST 5 - UNIBUS CONTENTION TEST
9.4 TEST 6 - MULTI-DRIVE INTERFERENCE TEST

10.0 ERROR REPORTING
10.1 COMMON ERRORS
10.2 ERROR HANDLING
10.3 ERROR PRINTOUT EXAMPLES

11.0 RK06 HEAD ALIGNMENT AID
11.1 HARDWARE REQUIREMENTS
11.2 OPERATIONAL MODES
11.2.1 MANUAL SELECT MODE
11.2.1.1 MANUAL SELECT ALIGNMENT
11.2.1.2 MANUAL SELECT VERIFY
11.2.1.3 MANUAL SELECT EXERCISE
11.2.2 AUTO SELECT MODE
11.2.2.1 AUTO SELECT ALIGNMENT
11.2.2.2 AUTO SELECT VERIFY
11.2.2.3 AUTO SELECT EXERCISE
11.3 ALIGNMENT AID ERROR MESSAGES

APPENDIX A SAMPLE ADDRESS 200 DEFAULT RUN

APPENDIX B SAMPLE ADDRESS 204 RUN

APPENDIX C SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN

APPENDIX D SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 2 OF THE RK611/RK06 SUBSYSTEM VERIFICATION TESTS, WHICH PROVIDE A FUNCTIONAL SHAKEDOWN OF THE ENTIRE SUBSYSTEM, INCLUDING THE UNIBUS INTERFACE AND ACCESS TO MAIN MEMORY. THE TESTING IN PART 2 EMPLOYS WORST-CASE SITUATIONS INVOLVING HEAD OFFSETTING, MEMORY ADDRESSING AND DATA TRANSFER, UNIBUS CYCLE CONTENTION, AND MULTIPLE DRIVE OPERATIONS. ADDITIONALLY, AN RK06 HEAD ALIGNMENT AID IS PROVIDED TO FACILITATE ON-LINE ALIGNMENT OF DRIVE HEADS.

WITHIN THE VARIOUS SUBSYSTEM TESTS, EMPHASIS IS GIVEN TO USEFUL SCOPE LOOPS AND OPERATOR SPECIFICATION OF TEST PARAMETER VALUES. AT THE BEGINNING OF TESTING, THE FOLLOWING OPTIONS MAY BE SPECIFIED BY THE USER:

- RK611 REGISTER ADDRESS
- RK06 VECTOR ADDRESS
- RK06 PRIORITY LEVEL
- DRIVE (S) TO BE TESTED
- TEST (S) TO BE RUN
- NUMBER OF TEST ITERATIONS
- INITIAL DISK ADDRESS ON TRANSFERS
- DATA PATTERNS USED
- STALL TIME BETWEEN OPERATIONS

IN ADDITION, STANDARD SWITCH REGISTER OPTIONS ARE PROVIDED, FOR A VARIETY OF PROGRAM LOOPING, RUNNING, AND REPORTING MODES.

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

2.1 REQUIREMENTS FOR SUBSYSTEM TESTS

THE FOLLOWING HARDWARE IS REQUIRED TO RUN PART 2 OF THE SUBSYSTEM VERIFICATION TESTS:

- PDP-11/04,(05,10 MFG. ONLY),20,34,35,40,45,50,70 OR PDQ
- 16 K MEMORY
- CONSOLE TELETYPE
- RK06 UNIBUS CONTROLLER (RK611)
- 1 TO 8 RK06 DRIVES
- 1 TO 8 RK06 DISK CARTRIDGES (FORMATTED IN 20 OR 22 SECTOR FORMAT)

2.2 REQUIREMENTS FOR HEAD ALIGNMENT AID

ADDITIONAL HARDWARE IS REQUIRED BY THE RK06 HEAD ALIGNMENT AID:

RK06 FIELD TEST BOX (OR ALIGNMENT SECTION THEREOF)
RK06 ALIGNMENT CARTRIDGE
RK06 HEAD ALIGNMENT TOOL

3.0 PRELIMINARY PROGRAM REQUIREMENTS

BEFORE RUNNING THE SUBSYSTEM VERIFICATION PROGRAM, THE RK611 CONTROLLER DIAGNOSTIC (MAINDEC-11-DZR6A THRU DZR6E AND DZR6K) AND RK06 DRIVE DIAGNOSTIC (MAINDEC-11-DZR6H THRU DZR6J) SHOULD FIRST BE RUN, TO RESOLVE BASIC, SOLID HARDWARE FAULTS.

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. SUBSYSTEM TESTS 1-6 MAY BE RUN IN CHAIN OR DUMP MODE, BUT THE ALIGNMENT AID MAY ONLY BE RUN IN DUMP MODE.

4.2.1 CHAIN MODE

THIS PROGRAM IS DESIGNED TO BE RUN IN THE DEFAULT MODE WHEN CHAINED UNDER XXDP. THUS, INPUT DIALOGUE IS BYPASSED, AND RK06 UNIBUS ADDRESS AND INTERRUPT VECTOR ARE DEFAULTED TO STANDARD VALUES. ALL DRIVES PRESENT AND READY WILL BE TESTED WITH THE EXCEPTION OF DRIVE 0, IF THE RK06 CONTAINS THE XXDP MEDIUM.

4.2.2 DUMP MODE

THE PROGRAM CAN BE RUN IN DUMP MODE, WITH OR WITHOUT DEFAULT PARAMETERS. DRIVE 0 MAY BE TESTED, BUT IF IT CONTAINS THE XXDP MEDIUM, THE OPERATOR MUST REPLACE THE XXDP PACK WITH A FORMATTED SCRATCH PACK PRIOR TO TESTING (OR AN ALIGNMENT CARTRIDGE IF ALIGNMENT IS TO BE DONE ON DRIVE 0). A MESSAGE WILL INFORM THE OPERATOR WHEN

THIS IS NECESSARY.

4.3 ACT/APT

THE PROGRAM IS COMPATIBLE WITH ACT/APT CONVENTIONS. SUBSYSTEM TESTS 1-6 MAY BE RUN IN AUTOMATIC OR DUMP MODE, AND THE HEAD ALIGNMENT AID MAY ONLY BE RUN IN DUMP MODE.

4.3.1 AUTOMATIC MODE

THE PROGRAM PROVIDES FOR AUTOMATIC APT/ACT OPERATION. IN THIS MODE, PARAMETERS REVERT TO DEFAULT VALUES, AND ALL DRIVES WHICH ARE PRESENT AND READY ARE TESTED.

4.3.2 DUMP MODE

IN DUMP MODE, PARAMETERS MAY BE INPUT, OR DEFAULTED, AND SUBSYSTEM TESTS 1-6 OR HEAD ALIGNMENT AID MAY BE RUN.

4.3.3 APT ETABLE DEFINITIONS

THE FOLLOWING DEFINITIONS ARE VALID FOR SPECIFYING APT ENVIRONMENTAL TABLE (ETABLE) ENTRIES, VIA THE APT UTILITY PROGRAM :

1. SOFTWARE ENVIRONMENT
   =1 IF APT SCRIPT MODE
   =0 IF STANDALONE MODE

2. ENVIRONMENT MODE BYTE
   BIT 7 = 1 ETABLE DOES SIZING
   = 0 PROGRAM DOES SIZING
   BIT 6 = 1 SPOOL MESSAGES TO APT IF SCRIPT MODE
   = 0 DON'T SPOOL TO APT
   BIT 5 = 1 SUPPRESS CONSOLE OUTPUT
   = 0 ALLOW CONSOLE OUTPUT
   BITS 4-0 NOT USED

3. SWITCH 1 (SOFTWARE SWITCH REGISTER)
IF ENVIRONMENT MODE BIT 7 (SIZING BIT) IS SET TO 1, THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD OF THE HARDWARE CONSOLE SWITCH REGISTER. REGARDLESS OF WHICH ONE IS USED, ALL BITS DEFINED IN SECTION 6.2 (SWITCH REGISTER OPTIONS) MAY USED WHEN RUNNING IN STANDALONE MODE.
IN APT SCRIPT MODE, HOWEVER, BIT 14 (LOOP ON TEST) MUST ALWAYS BE SET TO 0.

4. SWITCH 2 (USER SWITCH REGISTER)
NOT USED

5. MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES
NOT USED

6. INTERRUPT VECTOR 1
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 210

7. BUS PRIORITY 1
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 5

8. INTERRUPT VECTOR 2
NOT USED

9. BUS PRIORITY 2
NOT USED

10. BASE ADDRESS
USED WHEN ENVIRONMENT MODE BIT 7 = 1. DEFAULT = 177440

11. DEVICE MAP
USED WHEN ENVIRONMENT MODE BIT 7 = 1. EACH BIT SET TO 1 IN BITS 0-7 WILL SELECT THE CORRESPONDING DRIVE TO BE TESTED.
BITS 8-15 ARE NOT USED.

4.4 DUAL-ACCESS

THIS PROGRAM DOES NOT UTILIZE DUAL-ACCESS. FOR THE PURPOSES OF ALL TESTS (1-6), AND THE HEAD ALIGNMENT AID (SECTION 11), THE OPERATOR MUST GUARANTEE THAT THERE IS NO INTERFERENCE FROM THE UNUSED PORT. IF FAILURES ARE ENCOUNTERED IN TESTS 1-6 DUE TO INTERFERENCE FROM THE OTHER PORT, THE OPERATOR IS ADVISED TO SWITCH THAT PORT OFF-LINE AND RE-RUN THE TESTS.

4.5 MEMORY MANAGEMENT

THIS PROGRAM SUPPORTS MEMORY MANAGEMENT OPTIONS KT11C,D AND PDP11/70, FOR THE PURPOSES OF TESTS 2-4 ONLY. IN THESE TESTS (SEE SECTION 9.7) DIRECT ACCESS TO ALL OF PHYSICAL MEMORY ABOVE THE PROGRAM IS EXERCISED. FOR THE DURATION OF THESE 3 TESTS, MEMORY MANAGEMENT IS ENABLED. IN ALL OTHER TESTS, AND DURING THE USE OF THE HEAD ALIGNMENT AID, MEMORY MANAGEMENT IS DISABLED.

4.6 MEMORY PARITY CHECK

IF THE MEMORY PARITY CHECK OPTION IS AVAILABLE ON THE SYSTEM, ALL

TESTING IS DONE WITH PARITY CHECK ENABLED. PARITY ERRORS ARE REPORTED, AND TESTING IS ABORTED, UNLESS THE BYTE LABELED MEMABT WAS PREVIOUSLY LOADED WITH A NON-ZERO VALUE.

4.7 BAD SECTORS

ACCORDING TO A CONTROL SWITCH OPTION (SEE SECTION 8.2.4.2) THE BAD SECTOR FILE ON EACH DRIVE TO BE TESTED IS TYPED AT THE CONSOLE PRIOR TO TESTING. ALL DATA ERRORS OCCURRING IN THE PROGRAM ARE MASKED OUT IF THEY OCCUR ON SECTORS DESIGNATED AS BAD, (BY EITHER FACTORY OR SOFTWARE).

4.8 EXECUTION TIME

EXECUTION TIME IS DEPENDENT UPON PARAMETERS INPUT BY THE OPERATOR (SUCH AS STALL TIME OR ITERATION COUNT), AND TO A GREAT DEGREE, UPON THE AMOUNT OF MEMORY. HOWEVER, THE "AVERAGE TIME" REQUIRED TO RUN A QUICK VERIFICATION (FIRST PASS) IS 2 MINUTES (FOR A 64K SYSTEM). A SUBSEQUENT PASS WITH PARAMETERS DEFAULTED REQUIRES 3 MINUTES PER DRIVE (ON A 64-K SYSTEM).

5.0 PROGRAM LOADING

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

6.0 STARTING PROCEDURE

6.1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS OF TESTS 1-6 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS,RK06 UNIBUS ADDR. AND INTERRUPT VECTOR FOR TESTS 1-6
224 HEAD ALIGNMENT AID START ADDRESS

NOTE

FOR HEAD ALIGNMENT AID OPERATING INFORMATION PLEASE SKIP TO SECTION 11. THE SECTIONS WHICH IMMEDIATELY FOLLOW APPLY ONLY TO THE SUBSYSTEM TESTS (1-6).

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 TEST
13	INHIBIT ERROR REPORTS
12	REPORT DESCRIPTION ONLY, ON ERROR
11	INHIBIT ITERATIONS
10	BELL ON ERROR
09	LOOP ON ERROR
08	APPLY RANDOM STALL BETWEEN OPERATIONS
06	REPORT ONE ERROR PER TRANSFER IN TESTS 2-4
01	INHIBIT WRITES IN TEST 1
00	REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 2-4.

NOTE

FOR ADDITIONAL PROGRAM CONTROL OPTIONS, SEE DESCRIPTION OF CONTROL SWITCH WORD (CS), SECTION 8.2.4.2 .

7.0 OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 5.0)
2. LOAD A FORMATTED PACK ON EACH DRIVE TO BE TESTED.
3. BRING DRIVE (S) TO ONLINE STATE, WRITE ENABLED, AND LOCKED ON PORT.
4. LOAD THE DESIRED STARTING ADDRESS (SEE SECTION 6.1)
5. SET SWITCHES IF DESIRED (SEE SECTION 6.2)
6. PRESS START OR GIVE APPROPRIATE MONITOR START COMMAND.

8.0 PROGRAM ACTION

8.1 DESCRIPTION OF OPERATING PARAMETERS

AFTER THE PROGRAM IS STARTED, IT TYPES ITS IDENTIFICATION AS FOLLOWS: "DZR6N-C- RK11/RK06 SUBSYSTEM VERIFICATION:PART 2", FOLLOWED BY: "LAST PHYS MEM ADRS=XXXXXXXX". THEN, EITHER THE TESTS BEGIN EXECUTION

WITH DEFAULT PARAMETERS, OR AN OPERATOR INTERACTIVE MODE IS ENTERED (SEE STARTING ADDRESSES, SECTION 6.1). IN THIS MODE OPERATING PARAMETERS MAY BE LISTED OR ALTERED BY VARIOUS MEANS. ALL PARAMETERS ARE TREATED AS OCTAL NUMBERS, AND THE PARAMETER LIST IS SHARED AMONG THE TESTS.

THE FOLLOWING IS THE LIST OF OPERATING PARAMETERS, (WHICH APPLY TO THE CURRENT SELECTION OF DRIVES AND TESTS). AFTER EACH, IS INDICATED THE VALID RANGE OF VALUES FOR THAT PARAMETER (IN OCTAL), AND ITS DEFAULT VALUE. ALL PARAMETERS ARE ENTERED AS OCTAL NUMBERS, AND THE PARAMETER MNEMONICS SHOWN ARE THOSE USED BY THE PROGRAM.

FC — FIRST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS. FC=0-631, DEFAULT=0

LC — LAST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS. LC=0-632, DEFAULT=632

FT — FIRST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, FT=0-2, DEFAULT=0

LT — LAST TRACK ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, LT=0-2, DEFAULT=2

S0 — FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. S0=0-23, DEFAULT=0.

S1 — LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMAT CARTRIDGE. S1=0-23, DEFAULT=23.

S2 — FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE, S2=0-25, DEFAULT=0.

S3 — LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 22-SECTOR FORMATTED CARTRIDGE. S3=0-25, DEFAULT = 25.

PT — DATA PATTERN SELECT WORD (SEE SECTION 8.2.4.1 FOR DETAILS) PT=0-177777, DEFAULT=0

CS — CONTROL SWITCH WORD (SEE SECTION 8.2.4.2 FOR DETAILS) CS=0-000062, DEFAULT=0

ST — NUMBER OF UNIT STALL TIMES WITH WHICH TO STALL (DELAY) BETWEEN RK06 COMMANDS

8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)

8.2.1 DRIVE SELECTION

AFTER AN ADDRESS 204 START, THE PROGRAM INDICATES INPUT MODE BY TYPING "PARAMETER INPUT MODE". THEN, THE RK611 REGISTER ADDRESS, RK06 VECTOR ADDRESS AND RK06 PRIORITY ARE OPENED FOR POSSIBLE MODIFICATION AS FOLLOWS:

```
RK06 BUS ADR = 177440   NEW=(TYPE NEW VALUE HERE)
RK06 VEC ADR =    210   NEW=(TYPE NEW VALUE HERE)
RK06 PRIORITY=      5   NEW=(TYPE NEW VALUE HERE)
```

NEXT THE PROGRAM TYPES THE NUMBER(S) OF THE CURRENT DRIVES UNDER TEST, FOLLOWED BY A (*) ON THE NEXT LINE TO REQUEST NEW DRIVE NUMBERS. FOR EXAMPLE:

```
PARAMETER INPUT MODE

DRIVE(S)=0,1,2,4,7
* (INPUT, IF ANY, GOES HERE)
```

THE OPERATOR TYPES THE NEW NUMBERS (SEPARATED BY COMMAS) PLUS <CR> OR SIMPLY <CR> TO LEAVE THEM UNCHANGED. IF HE ENTERS DRIVE NUMBERS, THESE DRIVES ARE THEN CHECKED FOR VALID STATUS BY THE PROGRAM, AND ALL THOSE TYPED WHICH ARE ACCEPTABLE WILL NOW BE LISTED AS BEFORE. THE OPERATOR MAY MANUALLY RECONFIGURE THE DRIVES AND TYPE IN NEW DRIVE NUMBERS, AND THESE WILL BE CHECKED AND TYPED, UNTIL THE OPERATOR FINALLY TYPES JUST <CR>.

ON INITIAL ENTRY FROM ADDRESS 204, THE DRIVES WHICH ARE LISTED ARE ALL THOSE ON THE SUBSYSTEM WHICH ARE ON-LINE, READY, WRITE-ENABLED, AND LOADED WITH A CARTRIDGE OTHER THAN AN ALIGNMENT PACK. (THIS IS THE DEFAULT DRIVE SELECTION FOR ADDRESS 200 START). ON ALL OTHER ENTRIES TO THE DRIVE SELECTION ROUTINE (VIA <↑C> AS DESCRIBED IN SECTIONS 8.2.2.5 AND 8.2.3.6) THE PROGRAM FIRST TYPES :

```
TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>
* (CHARACTER GOES HERE)
```

NEXT, THE DRIVES ARE LISTED AS SHOWN ABOVE. IF (A) IS TYPED, ALL AVAILABLE DRIVES WILL BE LISTED, AND IF NOT, THE PREVIOUS SELECTION OF DRIVES WILL BE LISTED.

8.2.2 TEST SELECTION

WHEN THE OPERATOR FINALLY TYPES JUST <CR> IN RESPONSE TO THE DRIVE REQUEST, THE PROGRAM ENTERS THE TEST INPUT ROUTINE. THE FOLLOWING INSTRUCTIVE LINES ARE TYPED :

```
L = LIST TESTS
C = CHANGE TESTS
I = INPUT PARAMETERS AND RUN TESTS
```

THEN, THE FOLLOWING REQUEST IS MADE :

```
ENTER L,C, OR I
* (CHARACTER GOES HERE)
```

THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.

8.2.2.1 LIST TESTS, (L)

IF (L) IS TYPED, THE PROGRAM TYPES THE CURRENT DESIRED TEST LIST, AS FOLLOWS :

```
TEST     ITERATIONS
  1           0
  2      177777
  3         400
  4          25
  ETC.
```

THE ITERATION NUMBER IS THE NUMBER OF TIMES THE TEST WILL BE RUN ON THIS PASS, AND IT MUST BE BETWEEN 0 AND 177777. IF IT IS ZERO, THAT TEST WILL NOT BE RUN. ON THE FIRST TIME THROUGH, ALL TESTS ARE LISTED WITH DEFAULT ITERATIONS (YET TO BE DETERMINED).

WHEN THE LIST IS COMPLETED, THE PROGRAM RETURNS TO TYPE : "ENTER L,C, OR I" AGAIN.

8.2.2.2 CHANGE TESTS, (C)

IF (C) IS TYPED, THE PROGRAM THEN ASKS: "TO DEFAULT TESTS TYPE D<CR>, ELSE <CR>". IF JUST <CR> IS TYPED, THE PROGRAM SEQUENTIALLY OPENS EACH TEST FOR POSSIBLE ALTERATION OF THE ITERATION NUMBER. EACH TEST IS OPENED AND A NEW LINE IS TYPED, GIVING THE TEST NO. AND ITER. NO. IN OCTAL, FOLLOWED BY (*) ON THE SAME LINE. FOR EXAMPLE :

```
TEST     ITERATIONS
  0           0 * (INPUT, IF ANY, GOES HERE)
```

THE OPERATOR THEN TYPES THE NEW VALUE PLUS <CR>, OR JUST <CR> TO LEAVE IT UNCHANGED. IF THE OPERATOR TYPES A NUMBER FOLLOWED BY <!> (EXCLAMATION POINT), THE NUMBER JUST ENTERED WILL BE LOADED INTO THE ITERATION TABLE FOR ALL REMAINING TESTS. AFTER ALL TESTS HAVE BEEN OPENED, THE PROGRAM RETURNS TO TYPE : "ENTER L,C, OR I" AGAIN.

8.2.2.3 INPUT PARAMETERS AND RUN TESTS, (I)

IF (I) IS TYPED, THE PROGRAM ENTERS THE PARAMETER LIST ALTERATION

ROUTINE, DESCRIBED IN SECTION 8.2.3, IN WHICH OPERATING PARAMETERS MAY BE INPUT, AND TESTING BEGUN.

8.2.2.4 CONTROL Z (↑Z) FUNCTION

IF THE OPERATOR WISHES TO EXIT AT ANY TIME FROM L OR C MODE, AND RETURN TO SELECT A NEW MODE (L,C, OR I), CONTROL Z (↑Z) MAY BE TYPED.

8.2.2.5 CONTROL C (↑C) FUNCTION

IF THE OPERATOR WISHES TO TERMINATE L,C, OR I MODE, AND RETURN TO REQUEST NEW DRIVES AND TESTS, CONTROL C (↑C) MAY BE TYPED.

8.2.3 PARAMETER LIST ALTERATION

THE PROGRAM NEXT TYPES THE FOLLOWING INSTRUCTIVE LINES:

```
T = TYPE LIST
O = OPEN LIST
S = SET INDIVIDUAL PARAM.
R = RUN TESTS
ENTER T,O,S, OR R
* (CHARACTER GOES HERE)
```

THE OPERATOR TYPES THE DESIRED CHARACTER PLUS <CR>.

8.2.3.1 TYPE LIST, (T)

IF <T> IS TYPED, THE PROGRAM TYPES THE CURRENT PARAMETER LIST, AS FOLLOWS:

```
FC = XXXXXX
LC = XXXXXX
ETC.
```

WHEN THE LIST IS COMPLETED, THE PROGRAM RETURNS TO TYPE "ENTER T,O,S, OR R" AGAIN.

8.2.3.2 OPEN LIST, (O)

IF (O) IS TYPED THE PROGRAM THEN ASKS: "TO DEFAULT ALL PARAMETERS TYPE O<CR>, ELSE <CR>". IF JUST <CR> IS TYPED, THE PROGRAM SEQUENTIALLY OPENS EACH OPERATING PARAMETER IN THE LIST FOR ALTERATION

IN THE ORDER SHOWN IN SECTION 8.1). EACH PARAMETER IS OPENED AND A NEW LINE IS TYPED, GIVING THE PARAMETER AND ITS CURRENT VALUE (IN OCTAL), FOLLOWED BY (*) ON THE SAME LINE. FOR EXAMPLE:

```
IC=3 * (INPUT, IF ANY, GOES HERE)
```

THE OPERATOR THEN TYPES THE NEW VALUE PLUS A <CR> OR JUST <CR> TO LEAVE THAT PARAMETER UNCHANGED. AFTER ALL PARAMETERS HAVE BEEN OPENED FOR ALTERATION, THE PROGRAM RETURNS TO TYPE "ENTER T,O,S, OR R" AGAIN.

8.2.3.3 SET INDIVIDUAL PARAMETER, (S)

IF (S) IS TYPED, THE PROGRAM TYPES THE PROMPTER (>) ON THE NEXT LINE, AND WAITS FOR THE OPERATOR TO SET THE VALUE OF ANY INDIVIDUAL PARAMETER:

```
ENTER T,O,S, OR R
*S
> (ENTER PARAMETER AND VALUE HERE)
```

THE OPERATOR TYPES THE PARAMETER AND ITS NEW VALUE FOLLOWED BY <CR>, AND THE PROGRAM GIVES THE PROMPTER (>) AGAIN AS IN THE FOLLOWING EXAMPLE:

```
> FC = 600
> FS = 12
> IT = 1
> ETC.
```

TO TERMINATE INDIVIDUAL SETTING MODE, THE OPERATOR TYPES CONTROL Z (↑Z), AND THE PROGRAM RETURNS TO TYPE "ENTER T,O, S, OR R" AGAIN.

8.2.3.4 RUN TESTS, (R)

IF THE OPERATOR TYPES (R), EXECUTION OF TESTS BEGINS, USING ALL THE CURRENT PARAMETERS.

8.2.3.5 CONTROL Z (↑Z) FUNCTION

IF THE OPERATOR WISHES TO EXIT AT ANY TIME FROM T, O, S, OR R MODE (RUNNING TESTS), AND RETURN TO SELECT A NEW MODE (T,O,S, OR R), CONTROL Z (↑Z) MAY BE TYPED.

8.2.3.6 CONTROL C (↑C) FUNCTION

IF THE OPERATOR WISHES TO TERMINATE T, O, OR S MODE, OR STOP TEST EXECUTION (R MODE), CONTROL C (↑C) MAY BE TYPED, AND THE PROGRAM WILL RETURN TO REQUEST NEW DRIVE SELECTION (SEE SECTION 8.2.1). (ON AN ADDRESS 200 DEFAULT RUN, TYPING CONTROL C CAUSES THE PROGRAM TO HALT AFTER COMPLETING ANY DATA TRANSFER).

8.2.4 SPECIAL PARAMETER SPECIFICATIONS

8.2.4.1 PT-DATA PATTERN SELECT WORD

THE OPERATOR SPECIFIES A WORD OF SIXTEEN BITS (6 OCT. DIGITS), TO SELECT UP TO ALL SIXTEEN OF THE PATTERNS, LISTED IN SECTION 9.5.1. THE NO. OF ANY BIT SET IN PT CORRESPONDS TO THE NO. OF A PATTERN CHOSEN. IF THE HIGH BIT OF PT (BIT 15) IS SET, THE USER WILL BE ASKED TO TYPE UP TO SIXTEEN DATA WORDS WHICH WILL BE LOADED INTO THE BUFFER FOR PATTERN 15. THE PROGRAM TYPES:

"SELECT USER DEFINED PATTERN 15"

NEXT, EACH OF UP TO 16 WORDS ARE REQUESTED BY THE PROGRAM IN THE SAME MANNER IN WHICH THE OPERATING PARAMETERS ARE OPENED FOR ALTERATION (SEE SECT. 8.2.3.2):

```
WORD 0 = (OLD VALUE) * (NEW VALUE GOES HERE)
WORD 1 = (OLD VALUE) * (NEW VALUE GOES HERE)
ETC.
```

FOR EACH WORD, THE USER TYPES THE NEW VALUE PLUS <CR>, OR JUST <CR>, (TO LEAVE IT UNCHANGED). TO INPUT LESS THAN 16 WORDS FOR PATTERN 15, THE OPERATOR MAY TYPE (!) (EXCLAMATION POINT) AND THE PROGRAM WILL FILL UP THE REST OF THE PATTERN 15 BUFFER WITH THE LAST WORD SPECIFIED. NEXT, THE PROGRAM WILL CONTINUE, IN THE PROPER PARAMETER INPUT MODE (O OR S).

8.2.4.2 CS- CONTROL SWITCH WORD

IN ADDITION TO THE OPTIONS PROVIDED BY THE SWITCH REGISTER (HARDWARE OR SOFTWARE), A CONTROL SWITCH WORD (CS) MAY BE SPECIFIED AMONG THE OPERATING PARAMETERS TO PROVIDE THE FOLLOWING OPTIONS:

BIT	OPTION
05	DROP DRIVE IF 20 (DEC) ERRORS EXCEEDED
04	TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS)
01	INHIBIT OFFSET REPORTS IN TEST 1

NOTE

OTHER BITS UNUSED

8.3 DATA PATTERNS

THIS SECTION DESCRIBES THE DATA PATTERNS AVAILABLE FOR USE IN THE TESTS. EACH DATA PATTERN IS COMPRISED OF 16 WORDS. PATTERNS ARE REPEATED AS NECESSARY TO OBTAIN THE DESIRED WORD COUNT ON THE WRITE DATA COMMANDS. THE MAXIMUM ALLOWABLE WORD COUNT IS DETERMINED BY THE AMOUNT OF AVAILABLE BUFFER SPACE AT THE END OF THE PROGRAM, AND COULD BE AS LARGE AS 65,536 (DEC) WORDS. DATA SPIRALING AND TRACK SWITCHING IS THUS POSSIBLE DURING THE DATA TESTS.

THE FOLLOWING IS THE LIST OF 16 SELECTABLE DATA PATTERNS (IN OCTAL), WITH NOTABLE FEATURES DESCRIBED:

0 HIGH-LOW FREQUENCY MIX	1 HIGH FREQUENCY PHASE MIX	2 LOW FREQUENCY PHASE MIX	3 MAXIMUM PRECOMPENSATION PHASE MIX
177777	000000	052525	133333
177777	000000	052525	066666
177777	000000	052525	155555
052525	177777	125252	155555
052525	177777	125252	133333
052525	177777	125252	066666
177777	000000	052525	066666
177777	000000	052525	155555
052525	177777	125252	155555
052525	177777	125252	133333
177777	000000	052525	133333
052525	177777	125252	133333
177252	000000	052525	133333
177252	177777	125252	133333
172765	000000	052525	133333
172765	177777	125252	133333

4 ROTATING BOUNDARY PULSE PRECOMPENSATION	5 ROTATING CELL PULSE PRECOMPENSATION	6 ALL ZEROS	7 ALL ONES
121105	026455	000000	177777
150442	113226	000000	177777
064221	045513	000000	177777
132110	122645	000000	177777
055044	151322	000000	177777
026422	064551	000000	177777
013211	132264	000000	177777
105504	055132	000000	177777
042642	026455	000000	177777
021321	113226	000000	177777
110550	045513	000000	177777
044264	122645	000000	177777
022132	151322	000000	177777
011055	064551	000000	177777
104426	132264	000000	177777
042213	055132	000000	177777

8 SHIFTED 1'S IN FIELD OF 0'S	9 SHIFTED 0 IN FIELD OF 1'S	10 ALTERNATING 0-1	11 ALTERNATING 1-0
000001	177776	052525	125252
000002	177775	052525	125252
000004	177773	052525	125252
000010	177767	052525	125252
000020	177757	052525	125252
000040	177737	052525	125252
000100	177677	052525	125252
000200	177577	052525	125252
000400	177377	052525	125252
001000	176777	052525	125252
002000	175777	052525	125252
004000	173777	052525	125252
010000	167777	052525	125252
020000	157777	052525	125252
040000	137777	052525	125252
100000	077777	052525	125252

12 SHIFTING 0'S AND 1'S	13 COMPOSITE ROTATING	14 PSEUDO-RANDOM	15 USER-DEFINED
000001	072307		
000003	135143		
000007	156461		
000017	167230		
000037	073514		
000077	035646		
000177	016723		
000377	107351		
000777	143564		
001777	061672		
003777	030735		
007777	114356		
017777	046167		
037777	123073		
077777	151453		
177777	164616		

9.0 DESCRIPTION OF TESTS

9.1 TEST 1 - OFFSET-TO-FAILURE MEASUREMENT

IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC. THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH 167230(OCT) (TO ESTABLISH A KNOWN PATTERN), BUT THEY ARE NOT TESTED. THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE OFFSETS, UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER: SECTORS FS AND LS ON CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF INCREASING TRACKS, AS FOLLOWS:

OFFSET-TO-FAILURE MEASUREMENTS:

TRACK	CYLN	SECT	-OFST (UIN)	-OFST (UIN)
X	XXX	XX	XXXX	XXXX
X	XXX	XX	XXXX	XXXX
X	XXX	XX	XXXX	XXXX

ETC.

NOTE

IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS TO 12(OCT), FOR THIS TEST.

9.2 NPR / MAIN MEMORY TESTS

THIS GROUP OF TESTS EXERCISES ALL MEMORY ABOVE THE PROGRAM, VIA READ/WRITE NPR TRANSFERS WITH THE RK06. THE TESTS ARE DESIGNED TO DIAGNOSE MEMORY TRANSFER FAILURES DURING HEAVY NPR ACTIVITY. IN SYSTEMS PROVIDED WITH MEMORY MANAGEMENT (KT 11 C, D OR PDP 11/70), ALTHOUGH MEMORY MANAGEMENT IS NOT REQUIRED.

NOTE

THERE IS NO PROGRAM RELOCATION (EXCEPT FOR THE XXDP OR ABSOLUTE LOADER, IF

PRESENT). HENCE, MEMORY IN WHICH THE PROGRAM RESIDES IS NOT TESTED. ALSO, THE UNIBUS I/O ADDRESSES ARE NOT TESTED. IF IT IS DESIRED THAT THE LOADER BE DESTROYED, SO THAT ADDITIONAL MEMORY MAY BE TESTED, LOCATION 170 (KILLDR:) MAY MANUALLY BE SET TO A NON-ZERO VALUE.

THERE ARE 3 MEMORY TESTS PROVIDED. TESTS 2 AND 3 ARE TESTS OF MEMORY ADDRESSING CAPABILITY DURING DISK NPR'S, AND TEST 4 IS AN NPR/MEMORY DATA PATTERN TEST. DURING TESTING, MEMORY MANAGEMENT WILL BE ENABLED IF INSTALLED, AND ALL OPERATIONS WILL BE PERFORMED IN KERNAL MODE, WITH ADDRESS RELOCATION BEING DONE THROUGH MANIPULATION OF KERNAL I PAGE ADDRESS REGISTERS.

9.2.1 TEST 2 - NPR/MEMORY WORD ADDRESSING TEST

STARTING AT THE FIRST ADDRESS OF THE NEXT 1 K MEMORY BLOCK BEYOND THE END OF THE READ/WRITE DATA BUFFER (RWBUF). WRITE UNIQUE NUMBERS (STARTING WITH 1) INTO EACH OF UP TO 64K WORDS (DEPENDING ON THE AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES WITHIN THE 64K BLOCK. NEXT WRITE THE 64K WORDS (MAX) ONTO DISK AT FC, FS, FT (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW). THEN ZERO THE ENTIRE 64K BLOCK IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING VIRTUAL (IF MEM. MANAGEMENT) AND PHYSICAL ADDRESSES, AS WELL AS THE GOOD AND BAD DATA, FOR UP TO THE FIRST 10 (DECIMAL) FAILING LOCATIONS.

IF MEMORY MANAGEMENT IS PROVIDED AND THERE IS ADDITONAL MEMORY TO TEST, REPEAT THE ABOVE ADDRESSING TEST FOR EACH OF THE REMAINING 64K PHYSICAL MEMORY BLOCKS.

9.2.2 TEST 3 - NPR/MEMORY BLOCK ADDRESSING TEST

IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06 NPR'S IS TESTED.

THE TEST BEGINS BY WRITING A SINGLE NUMBER INTO ALL LOCATIONS OF EACH MEMORY BLOCK OF UP TO 64K WORDS. THIS NUMBER UNIQUELY DEFINES EACH BLOCK. THEN WRITE EACH BLOCK ONTO DISK AT ADDRESS FC, FS, FT (SCALED, IF NECESSARY TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD SECTOR FILE). ZERO THE BLOCK IN MEMORY, AND THEN READ THE DATA BACK INTO THE PROPER PHYSICAL ADDRESSES, DO THIS FOR ALL 64K BLOCKS, AND THEN COMPARE THE CONTENTS OF ALL THE MEMORY TO EXPECTED VALUES. TYPE FAILURES AS DESCRIBED ABOVE FOR THE FIRST 10(DEC) FAILING LOCATIONS IN EACH 64K MEMORY BLOCK.

9.2.3 TEST 4 - NPR/MEMORY DATA PATTERN TEST

IN THIS TEST, ALL PHYSICAL MEMORY LOCATIONS ABOVE THE PROGRAM (NOT INCLUDING UPPER UNIBUS DEVICE ADDRESSES) ARE EXERCISED WITH UP TO 15 DATA PATTERNS, AS CHOSEN FROM THE FIRST 14 PATTERNS DESCRIBED IN SECTION 8.3, PLUS PATTERN 15 (USER DEFINED PATTERN). THE DATA PATTERNS DESIRED FOR THIS TEST ARE CHOSEN SEPARATELY, HOWEVER, AND THEY DEFAULT TO PATTERNS 8, 9, 10, AND 11.

MEMORY IS TESTED IN BLOCKS OF 64K, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE PROGRAM. EACH BLOCK OF UP TO 64K IS LOADED WITH DATA, WRITTEN ONTO DISK AT FC, FS, FT, LOADED WITH ZEROS, AND READ BACK AND COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN, INCLUDING AN OPTIONAL PATTERN CHOSEN BY THE USER.

9.3 TEST 5 - UNIBUS CONTENTION TEST

THE PURPOSE OF THIS TEST IS TO EXERCISE THE NPR CONTROL LOGIC WITHIN THE CONTROLLER IN SUCH A MANNER THAT A CONTENTION FOR MEMORY CYCLES ARISES BETWEEN THE PROCESSOR AND THE RK06.

FIRST, A WRITE DATA IS BEGUN ON CYLINDER FC, (SCALED TO AVOID PACK OVERFLOW) AT SECTOR 0, WITH THE BUS ADDRESS INCREMENT INHIBIT BIT SET IN THE CONTROLLER. USING A WORD COUNT OF 12000(OCT) OR 13000(OCT) (DEPENDING ON THE FORMAT), THE ENTIRE TRACK IS WRITTEN WITH THE FIRST WORD OF USER DEFINED PATTERN 15 (IF SELECTED) OR THE FIRST WORD OF PATTERN 13 (BY DEFAULT). THEN, AN INSTRUCTION LOOP IS ENTERED IN WHICH A NON-EXISTENT UNIBUS DEVICE ADDRESS IS REFERENCED ONCE PER PASS. THIS WILL RESULT IN A PROCESSOR SIEZURE OF THE UNIBUS, FOR 5-20 MICRO-SEC (DEPENDING ON THE PROCESSOR), FOLLOWED BY A NON-EXISTENT MEMORY PROCESSOR TRAP, ONCE PER PASS. THUS, FOR EACH PASS THROUGH THE INSTRUCTION LOOP, THE CONTROLLER IS FORCED TO LOSE FROM ONE TO FOUR NPR BUS CYCLES. THE EXECUTION TIME OF THE LOOP DETERMINES THE REPITITION RATE OF THE NON-EXISTENT MEMORY REFERENCES, AND THIS IS CHOSEN TO ALLOW ENOUGH NPR'S SO THAT THERE SHOULD BE NO DATA LATE ERRORS IN A FAULT-FREE CONTROLLER. THE DATA IS THEN WRITE CHECKED FOR VERIFICATION, AND THE ABOVE TEST IS REPEATED USING A READ DATA COMMAND. NO ERRORS ARE EXPECTED.

THE ABOVE OPERATIONS ARE REPEATED 15 MORE TIMES, USING THE NEXT WORD OF THE PROPER PATTERN (13 OR 15) EACH TIME, TO WRITE AND READ THE ENTIRE TRACK.

9.4 TEST 6 - MULTI-DRIVE INTERFERENCE TEST

THIS TEST PERFORMS MULTI-DRIVE POSITIONING OPERATIONS, WHILE A LARGE DATA TRANSFER IS IN PROGRESS ON THE DRIVE UNDER TEST, FOR THE PURPOSE OF DETECTING PROBLEMS OF CONCURRENT DRIVE OPERATION. THE TEST IS RUN ONLY IF THERE IS MORE THAN ONE DRIVE ON THE SUBSYSTEM.

THE TEST PROCEEDS AS FOLLOWS: IT IS FIRST DETERMINED WHICH DRIVE(S) (BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH. NEXT, A SEEK IS DONE TO CYLINDER FC (SCALED, IF NECESSARY) ON THE DRIVE UNDER TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH TO A DIFFERENT PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE FROM 10-67 MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN, A WRITE DATA IS BEGUN ON THE DRIVE UNDER TEST, AT THE CURRENT CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616 (DEC) WORDS IF 20 SECTOR FORMAT, OR 17,152 (DEC) WORDS IF 22 SECTOR FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA. THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN THAT TIME, ALL THE OTHER DRIVES SHOULD HAVE COMPLETED POSITIONING. NEXT, A WRITE CHECK IS DONE TO VERIFY THE DATA WRITTEN ON THE DRIVE UNDER TEST, AND A CHECK IS MADE TO INSURE THAT ALL OTHER DRIVES WHICH PERFORMED SEEKS WERE ABLE TO COMPLETE THEM SUCCESSFULLY, BY CHECKING ATTENTION BITS AND COMPARING DRIVE CYLINDER ADDRESSES TO EXPECTED VALUES.

10.0 ERROR REPORTING

10.1 COMMON ERRORS

THE FOLLOWING IS A LIST OF COMMON ERROR MESSAGES WHICH ACCOMPANY ERROR TYPEOUTS FROM THE SUBSYSTEM VERIFICATION 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 - UFL 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

10.2 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. IN THE CASE OF A DATA MISCOMPARE, FOR INSTANCE, THE NUMBER GOOD DATA WORD, BAD DATA WORD, PHYSICAL MEMORY ADDRESS, VIRTUAL ADDRESS, AND MEMORY MANAGEMENT REGISTER CONTENTS (IF PRESENT) ARE REPORTED. OTHER ERROR REPORTS MAY CONSIST OF A SINGLE DESCRIPTIVE LINE.

10.3 ERROR PRINTOUT EXAMPLES

EXAMPLE 1:

```
     **  WRITE CHECK ERROR
         TEST 2

     PREVIOUS COMMAND:
     DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
     000000  000121  000016  000001  000000  175000
```

```
1216
1217
1218
1219                                        HI BA   LO BA
1220                                        000000  061566
1221
1222                                      CURRENT COMMAND:
1223                                      ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
1224                                      041154  000000  000131  000016  000001  000006  175000
1225                                        HI BA   LO BA
1226                                        000000  061566
1227
1228                                      PACK ADDRESS OF ERROR(S):
1229                                      CYLNDR  TRACK   SECTOR
1230                                      000016  000001  000006
1231
1232
1233                                   EXAMPLE 2:
1234
1235                                        **  DATA MISCOMPARE
1236                                            TEST 2
1237
1238                                        PREVIOUS COMMAND:
1239                                        DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
1240                                        000000  000131  000016  000001  000000  175000
1241                                          HI BA   LO BA
1242                                          000000  074000
1243
1244                                        CURRENT COMMAND:
1245                                        ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
1246                                        033156  000000  000121  000016  000001  000000  175000
1247                                          HI BA   LO BA
1248                                          000000  074000
1249
1250                                        PACK ADDRESS OF ERROR(S):
1251                                        CYLNDR  TRACK   SECTOR
1252                                        000016  000001  000000
1253
1254                                        WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6
1255
1256                                        000400  000401  125252  000000  075000  155000  000600
1257                                        000401  000402  125252  000000  075002  155002  000600
1258                                        000402  000403  125252  000000  075004  155004  000600
1259                                        000403  000404  125252  000000  075006  155006  000600
1260                                        000404  000405  125252  000000  075010  155010  000600
1261                                        000405  000406  125252  000000  075012  155012  000600
1262                                        000406  000407  125252  000000  075014  155014  000600
1263                                        000407  000410  125252  000000  075016  155016  000600
1264                                        000410  000411  125252  000000  075020  155020  000600
1265                                        000411  000412  125252  000000  075022  155022  000600
1266
1267
1268
1269                                   11.0  RK06 HEAD ALIGNMENT AID
1270
1271                                   THIS PROGRAM IS PROVIDED AS A SOFTWARE AID TO HEAD  ALIGNMENT  OF  THE
```

RK06 DRIVE, TO BE USED IN CONJUNCTION WITH THE RK06 FIELD TEST BOX, OR OTHER SUITABLE INDICATOR OF HEAD ALIGNMENT. THE PROGRAM OPERATES IN TWO MODES. THE FIRST IS MANUAL SELECT MODE, WHICH SELECTS A SPECIFIC DRIVE AND HEAD TO BE ALIGNED BY TTY INPUT AT THE CONSOLE. THE SECOND MODE IS AUTO-SELECT MODE, WHICH ALLOWS REMOTE DRIVE AND HEAD SELECTION THROUGH OPERATION OF DRIVE DUAL-PORT SELECT SWITCHES. IN EITHER MODE THE OPERATOR HAS THE OPTION TO EITHER PERFORM HEAD ALIGNMENT, VERIFY ALIGNMENT, OR REQUEST UP TO FIVE MINUTES OF RANDOM SEEK EXERCISES TO BE PERFORMED ON THE SPECIFIED DRIVE (S).

11.1 HARDWARE REQUIREMENTS

TO PERFORM HEAD ALIGNMENT, THE ALIGNMENT INDICATING DEVICE MUST BE CONNECTED TO THE DRIVE VIA THE ALIGNMENT CABLE WHICH ATTACHES TO THE RK06 READ/WRITE MODULE. THE ALIGNMENT INFORMATION FROM THE DRIVE MUST BE DISPLAYED ON A METER OR OTHER INDICATOR. EACH DRIVE TO BE ALIGNED MUST BE LOADED WITH AN RK06 ALIGNMENT CARTRIDGE, AND MUST BE WRITE-PROTECTED. UP TO EIGHT DRIVES PER CONTROLLER MAY BE ALIGNED, AND EACH DRIVE MAY BE OPERATED IN SINGLE-PORT MODE ONLY. FOR THE PURPOSE OF THE HEAD ALIGNMENT AND ALL SWITCH REGISTER BITS (HARDWARE OR SOFTWARE) ARE IGNORED.

11.2 OPERATIONAL MODES

THE PROGRAM COMMUNICATES WITH THE OPERATOR DOING THE ALIGNMENT THROUGH CONSOLE DIALOGUE. WHEN THE PROGRAM IS STARTED AT ADDRESS 224(OCTAL), IT TYPES IDENTIFICATION:

```
" *** RK06 HEAD ALIGNMENT AID ***
TYPE H(CR) FOR INFORMATION- OTHERWISE, TYPE (CR) "
```

NEXT, THE PROGRAM REQUESTS THE DESIRED MODE OF OPERATION:

```
" MANUAL OR AUTO MODE (M OR A) ?  "
```

THE OPERATOR MAKES THE SELECTION, AND ENTERS ONE OF THE FOLLOWING MODES.

11.2.1 MANUAL SELECT MODE

IN THIS MODE, THE DRIVE(S) TO BE ALIGNED AND THE HEAD(S) TO BE SELECTED ARE SPECIFIED BY TTY INPUT. THIS MODE IS USEFUL FOR SYSTEMS WITH FEW DRIVES, WHICH ARE IN CLOSE PROXIMITY TO THE CONSOLE, AS IN TYPICAL FIELD INSTALLATIONS.

THE PROGRAM FIRST ECHOS THE MODE:

"* MANUAL SELECT MODE * "
AND TYPES "ENTER DRIVE NO. (0-7):".

THE OPERATOR TYPES THE DRIVE NUMBER, THE PROGRAM TYPES THE DRIVE SERIAL NUMBER, AND THEN ASKS "ALIGN, VERIFY, OR EXERCISE (A, V, OR E: "". THE OPERATOR RESPONDS, AND THE PROGRAM ENTERS THE APPROPRIATE SUB-MODE.

11.2.1.1 MANUAL SELECT ALIGNMENT

ALIGNMENT IN MANUAL MODE PROCEEDS AS FOLLOWS. THE PROGRAM TYPES:

"* MANUAL SELECT ALIGNMENT *," FOLLOWED BY
" ENTER HEAD NO. (0-2):"

THE OPERATOR TYPES THE DESIRED HEAD NUMBER, AND THE PROGRAM UNLOADS THE HEADS ON THE DRIVE, IN ANTICIPATION OF OPERATOR MOUNTING OF THE HEAD ALIGNMENT TOOL. THE PROGRAM TYPES "TYPE <R> WHEN READY:". AFTER THE TOOL HAS BEEN MOUNTED THE OPERATOR TYPES <R>, AND THE PROGRAM LOADS HEADS AND RESPONDS WITH:

"HEADS POSITIONED AT CYLINDER 365.(OCT)"
AND "HEAD X SELECTED"

THE POSITIONING TO CYLINDER 365 IS DONE IN SINGLE INCREMENT SEEKS, TO AVOID EXCESSIVE MOMENTUM ON THE POSITIONER, WHILE THE ALIGNMENT TOOL IS INSTALLED. THE OPERATOR MAY NOW PROCEED TO ALIGN THIS HEAD, AND THE PROGRAM LOOPS BACK TO ASK FOR ANOTHER HEAD NUMBER ON THIS DRIVE. IF THE OPERATOR TYPES CONTROL Z (↑Z) THE PROGRAM WILL LOOP BACK TO ASK FOR "ALIGN OR EXERCISE ?" (STILL IN THE SAME MODE). IF CONTROL C (↑C) IS TYPED, THE PROGRAM LOOPS FURTHER BACK TO ASK FOR A NEW DRIVE NUMBER (STILL IN THE SAME MODE). IF CONTROL R (↑R) IS TYPED, THE PROGRAM EXITS FROM THE CURRENT MODE, AND DOES A COMPLETE RESTART.

11.2.1.2 MANUAL SELECT VERIFY

VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

"* MANUAL SELECT VERIFY *" , FOLLOWED BY
"ENTER HEAD NO. (0-2) :"

THE VERIFY MODE IS IDENTICAL TO THE ALIGNMENT MODE, EXCEPT THAT UNLOADING AND LOADING OF HEADS IS NOT DONE BETWEEN HEAD SELECTIONS. THEREFORE, IN THIS MODE, THE OPERATOR MAY SELECT HEADS FOR QUICK ALIGNMENT VERIFICATION (USING THE ALIGNMENT METER), WITHOUT THE DELAYS OF UNLOADING AND LOADING HEADS. IN THIS MODE, THE OPERATOR WILL NOT BE ASKED TO "TYPE <R> WHEN READY".

11.2.1.3 MANUAL SELECT EXERCISE

WHEN THE EXERCISE SUB-MODE OF MANUAL SELECT MODE IS ENTERED, THE PROGRAM UNLOADS HEADS ON THE SELECTED DRIVE, AND TYPES "TYPE <R> WHEN READY". THE OPERATOR MUST THEN REMOVE THE ALIGNMENT TOOL PRIOR TO THE RANDOM SEEKS WHICH FOLLOW, AND ALLOW EXERCISES TO PROCEED BY TYPING <R>. THE PROGRAM THEN LOADS HEADS, AND RESPONDS WITH:

"* SEEK EXERCISES IN PROGRESS ON DRIVE X *".

AT THIS POINT, 7500 RANDOM SEEKS ARE BEGUN ON THE DRIVE, WHICH LASTS FOR ABOUT FIVE MINUTES. UPON COMPLETION OF THESE EXERCISES, THE PROGRAM RETURNS TO ASK FOR "ALIGN OR EXERCISE ?" (STILL IN MANUAL MODE) ON THIS DRIVE. WHILE IN EXERCISE SUB-MODE, THE CHARACTERS (^Z), (^C), AND (^R) PERFORM THE SAME FUNCTIONS AS DESCRIBED IN SECTION 11.2.1.1.

11.2.2 AUTO SELECT MODE

IN THIS MODE, ALL SELECTION OF DRIVES AND HEADS FOR ALIGNMENT IS DONE BY SEQUENTIAL OPERATION OF DRIVE PORT SELECT SWITCHES. THIS ALLOWS COMMUNICATION WITH THE PROGRAM TO BE REMOTE FROM THE CONSOLE. THUS AUTO SELECT MODE IS WELL SUITED TO THE MANUFACTURING TEST ENVIRONMENT, OR TO A FIELD INSTALLATION, WHICH HAS A NUMBER OF REMOTE DRIVES. THE PROGRAM FIRST ECHOS THE MODE:

" * AUTO SELECT MODE * ", AND ASKS
"ALIGN OR EXERCISE (A OR E) ?".

THE OPERATOR RESPONDS, AND THE PROGRAM ENTERS THE APPROPRIATE SUB-MODE.

11.2.2.1 AUTO SELECT ALIGNMENT

ALIGNMENT IN AUTO SELECT MODE PROCEEDS AS FOLLOWS. THE PROGRAM TYPES:

" * AUTO SELECT ALIGNMENT * ".

THE PROGRAM CONSTANTLY SCANS ALL DRIVES 0-7 TO DETERMINE THE EXISTENCE OF EACH. WHENEVER A DRIVE IS DESELECTED BY THE OPERATION OF ITS PORT SELECT SWITCH, THE PROGRAM RECEIVES A NON-EXISTENT DRIVE INDICATION WHEN IT SELECTS THAT DRIVE. THEN, WHEN THE PROGRAM DETERMINES THAT A DRIVE WHICH WAS PREVIOUSLY OFF-LINE (DESELECTED) HAS JUST BECOME ON-LINE (SELECTED), IT PREPARES FOR ALIGNMENT ON THAT DRIVE. IN SUMMARY, THE OPERATOR SELECTS A DRIVE FOR ALIGNMENT BY DEPRESSING AND THEN RELEASING THE PORT SELECT SWITCH (FOR THE PORT IN USE) ON THAT DRIVE.

THE PROGRAM RECOGNIZES THE DRIVE SELECTED, AND CHECKS IT TO BE SURE

THAT IT IS WRITE-LOCKED. THE PROGRAM THEN TYPES:

"DRIVE X SELECTED"

AND THEN IT UNLOADS THE HEADS ON THAT DRIVE, SO THAT THE OPERATOR CAN MOUNT THE HEAD ALIGNMENT TOOL. WHEN IT IS MOUNTED THE OPERATOR MUST DEPRESS AND RELEASE THE PORT SWITCH AGAIN, AND THE PROGRAM LOADS HEADS, SEEKS IN SINGLE INCREMENTS OUT TO THE ALIGNMENT CYLINDER (365 OCT.), AND TYPES:

"HEADS POSITIONED AT CYLINDER 365(OCT.)
 HEAD 0 SELECTED".

AFTER THE OPERATOR ALIGNS HEAD 0, HE MAY PROCEED TO HEAD 1, BY DEPRESSING AND RELEASING THE PORT SELECT SWITCH. WHEN HE DOES, THE HEADS WILL BE UNLOADED AGAIN, AND THE OPERATOR MAY MOVE THE ALIGNMENT TOOL TO HEAD 1, AND UPON DEPRESSING AND RELEASING THE PORT SWITCH AGAIN, THE HEADS WILL BE LOADED, THE PROGRAM WILL SEEK TO CYLINDER 365(OCT), AND TYPE:

"HEADS POSITIONED AT CYLINDER 365(OCT)
 HEAD 1 SELECTED".

THIS PROCESS MAY BE CONTINUED WITH HEADS BEING SELECTED IN THE ORDER 0, 1, 2, 0, 1, 2, 0, ETC. UNTIL THE OPERATOR HAS COMPLETED THE ALIGNMENT.

WHEN THE OPERATOR COMPLETES ALIGNMENT ON THIS DRIVE, HE SHOULD REMOVE THE ALIGNMENT TOOL, AND SWITCH THE DRIVE ON-LINE. IF HE THEN WISHES TO SELECT ANOTHER DRIVE, HE MUST SWITCH THE DESIRED DRIVE OFF-LINE AND THEN ON-LINE AGAIN TO INITIATE SELECTION. WHEN ALL DESIRED DRIVES HAVE BEEN ALIGNED THE OPERATOR TYPES ↑Z OR ↑R, AS DESCRIBED IN SECTION 11.2.1.1. BEFORE LEAVING ALIGNMENT MODE AND PROCEEDING TO DO SEEK EXERCISES, THE OPERATOR MUST BE SURE TO REMOVE THE ALIGNMENT TOOL(S) FROM ALL DRIVE(S).

NOTE

THE ↑C FUNCTION DOES NOT APPLY IN AUTO MODE.

11.2.2.2 AUTO SELECT VERIFY

VERIFY MODE IDENTIFIES ITSELF AS FOLLOWS :

"* AUTO SELECT VERIFY *"

THE AUTO SELECT VERIFY MODE IS IDENTICAL TO THE AUTO ALIGNMENT MODE.

EXCEPT THAT UNLOADING AND LOADING OF HEADS IS NOT DONE BETWEEN HEAD SELECTIONS. THEREFORE, IN THIS MODE, THE OPERATOR MAY SELECT HEADS FOR QUICK ALIGNMENT VERIFICATION (USING THE ALIGNMENT METER), WITHOUT THE DELAY OF UNLOADING AND LOADING HEADS.

11.2.2.3 AUTO SELECT EXERCISE

IN AUTO SELECT EXERCISE SUB MODE, RANDOM SEEK EXERCISES ARE PERFORMED UPON ALL DRIVES WHICH ARE ON-LINE. DRIVES ARE EXERCISED SEQUENTIALLY, STARTING WITH DRIVE 0. THE OPERATOR SPECIFIES EITHER LONG EXERCISES (7500 SEEKS- 5 MINUTES) OR SHORT EXERCISES (1500 SEEKS- 1 MINUTE) TO BE PERFORMED UPON EACH DRIVE.

AUTO SELECT EXERCISE MODE IDENTIFIES ITSELF AS FOLLOWS:

" * AUTO SELECT EXERCISES * "
FOLLOWED BY"SHORT OR LONG (S OR L) ?".

THE OPERATOR TYPES HIS RESPON_ , AND THE SELECTED SEEK EXERCISES ARE PERFORMED ON EACH DRIVE. UPON COMPLETION OF THESE EXERCISES, THE PROGRAM RETURNS TO ASK FOR "ALIGN OR EXERCISE?" (STILL IN AUTO MODE). WHILE IN EXERCISE SUB-MODE THE CHARACTERS ↑Z AND ↑R PERFORM THE SAME FUNCTIONS AS DESCRIBED IN SECTION 11.2.1.1.

NOTE

BEFORE REQUESTING AUTO-EXERCISE MODE, THE OPERATOR MUST REMOVE THE ALIGNMENT TOOL (WHILE STILL IN AUTO-ALIGNMENT MODE) WHILE THE HEADS ARE UNLOADED.

11.3 ALIGNMENT AID ERROR MESSAGES

1. "DRIVE X NOT READY! PLEASE START IF DESIRED, THEN PRESS 'CONT' ON CPU WHEN READY." - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS FOUND NOT READY.

2. "DRIVE X NOT WRITE-LOCKED! PLEASE SET WRITE LOCK SWITCH. PRESS 'CONT' ON CPU WHEN READY". - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS FOUND NOT TO BE WRITE-PROTECTED.

3. "PLEASE LOAD ALIGNMENT CARTRIDGE ON DRIVE X! PRESS 'CONT' ON

CPU WHEN READY." - THIS INDICATES THAT A DRIVE SELECTED FOR ALIGNMENT WAS NOT LOADED WITH AN ALIGNMENT CARTRIDGE.

4. "MULTIPLE DRIVES AUTO-SELECTED! PRESS 'CONT' TO RESTART." - THIS INDICATES THAT IN AUTO ALIGNMENT MODE MORE THAN ONE OF THE DRIVES WERE SELECTED FOR ALIGNMENT SIMULTANEOUSELY.

5. "CANNOT READ BAD SECTOR FILE ON DRIVE X! PRESS 'CONT' ON CPU TO RESTART". - THIS INDICATES THAT A READ ERROR WAS ENCOUNTERED WHILE THE PROGRAM WAS ATTEMPTING TO IDENTIFY THE CARTRIDGE ON DRIVE X AS AN ALIGNMENT CARTRIDGE.

6. "? X" - THIS IS TYPED BY THE PROGRAM IN RESPONSE TO THE INPUT OF AN INVALID PARAMETER, SHOWN HERE AS X. THE OPERATOR SHOULD NOW ENTER THE PROPER VALUE.

APPENDIX A

SAMPLE ADDRESS 200 DEFAULT RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION: PART 2, ON A SINGLE DRIVE (DRIVE 0), FOR THE FIRST PASS OF THE PROGRAM (QUICK-VERIFY PASS). IN THIS MODE, ALL MESSAGES APPEARING HERE WERE OUTPUTS--NO OPERATOR INPUTS ARE PROVIDED ON AN ADDRESS 200 RUN.

```
DZR6N-C - RK611/RK06 SUBSYSTEM VERIFICATION :  PART 2

LAST PHYS MEM ADR = 377776

DRIVE 1 NON-EXISTENT
DRIVE 2 NON-EXISTENT
DRIVE 3 NON-EXISTENT
DRIVE 4 NON-EXISTENT
DRIVE 5 NON-EXISTENT
DRIVE 6 NON-EXISTENT
DRIVE 7 NON-EXISTENT

DRIVE(S) = 0

TESTING DRIVE 0
DRIVE SER. NO.  8
CART. SER. NO.  334


OFFSET-TO-FAILURE MEASUREMENTS:

TRACK  CYLN  SECT  +OFST  -OFST
                   (UIN)  (UIN)
  0      0     0   NONE   NONE
  0      0    12   NONE   1175
  0    631     0   NONE   NONE
  0    631    12   NONE   NONE
  1      0     0   1050   NONE
  1      0    12   NONE   1200
  1    631     0   NONE   NONE
  1    631    12   NONE   NONE
  2      0     0   NONE   NONE
  2      0    12   NONE   NONE
  2    631     0   NONE   NONE
  2    631    12   NONE   NONE

END PASS #   1
```

APPENDIX B

SAMPLE ADDRESS 204 RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART, 2, ON A SINGLE DRIVE (DRIVE 0), WITH PARTICULAR TESTS SELECTED TO BE RUN FOR SPECIFIED NUMBERS OF TIMES, WITH SPECIFIED PARAMETERS, AND A SPECIFIED NO. OF PROGRAM PASSES. SPECIFICALLY, IT WAS DESIRED THAT TEST 1 BE RUN ONCE, TEST 4 ONCE, AND TEST 6 ONCE. THESE TESTS WERE SPECIFIED TO RUN FOR 1 PROGRAM PASS, USING PARAMETERS SPECIFIED BY THE OPERATOR. IN THIS CASE, SECTOR ADDRESS LIMITS (S2 AND S3) AND DATA PATTERN (PT) WERE SPECIFIED AS NON-DEFAULT VALUES.

```
DZR6N-C - RK611/RK06 SUBSYSTEM VERIFICATION:PART 2


LAST PHYS MEM ADR=377776


PARAMETER INPUT MODE

RK06 BUS ADR  = 177440  NEW =
RK06 VEC ADR  =    210  NEW =
RK06 PRIORITY =      5  NEW =


DRIVE(S) = 0
*

L = LIST TESTS
C = CHANGE TESTS
I = INPUT PARAMETERS AND RUN TESTS

ENTER L,C, OR I
* C
TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>
*

TEST    ITERATIONS
  1        2 * 1
  2        2 * 0
  3        2 * 0
  4        2 * 0\0\1
```

```
1684
1685
1686
1687                                    5       2 * 0
1688                                    6     100 * 1
1689
1690                                ENTER L,C, OR I
1691                                * L
1692
1693                                TEST    ITERATIONS
1694                                  1         1
1695                                  2         0
1696                                  3         0
1697                                  4         1
1698                                  5         0
1699                                  6         1
1700
1701                                ENTER L,C, OR I
1702                                * I
1703
1704                                T = TYPE PARAMETER LIST
1705                                O = OPEN PARAMETER LIST
1706                                S = SET INDIVIDUAL PARAM
1707                                R = RUN TESTS
1708
1709                                ENTER T,O,S, OR R
1710                                * T
1711                                FC=0
1712                                LC=632
1713                                FT=0
1714                                LT=2
1715                                S0=0
1716                                S1=23
1717                                S2=0
1718                                S3=25
1719                                PT=0
1720                                CS=0
1721                                ST=0
1722
1723                                ENTER T,O,S, OR R
1724                                * S
1725                                > S2=4
1726                                > SV=20
1727                                SV=20?
1728                                > S3=20
1729                                > PT=100000
1730                                TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR>
1731                                * M
1732
1733                                MODIFY USER-DEFINED PATTERN 15:
1734                                WORD 00 = 072307 * 123456!
1735                                > ↑Z
1736
1737                                ENTER T,O,S, OR R
1738                                * T
1739                                FC=0
```

```
1740
1741
1742
1743                                    LC=632
1744                                    FT=0
1745                                    LT=2
1746                                    S0=0
1747                                    S1=23
1748                                    S2=4
1749                                    S3=20
1750                                    PT=100000
1751                                    CS=0
1752                                    ST=0
1753
1754                                    USER-DEFINED PATTERN 15:
1755                                    WORD 00 = 123456
1756                                    WORD 01 = 123456
1757                                    WORD 02 = 123456
1758                                    WORD 03 = 123456
1759                                    WORD 04 = 123456
1760                                    WORD 05 = 123456
1761                                    WORD 06 = 123456
1762                                    WORD 07 = 123456
1763                                    WORD 10 = 123456
1764                                    WORD 11 = 123456
1765                                    WORD 12 = 123456
1766                                    WORD 13 = 123456
1767                                    WORD 14↑Z
1768                                     = 123456
1769
1770                                    ENTER T,O,S, OR R
1771                                    * R
1772
1773                                    ENTER NO. OF PASSES (1-77777) :
1774                                    * 1
1775
1776                                    TESTING DRIVE 0
1777                                    DRIVE SER. NO.  8
1778                                    CART. SER. NO.  334
1779
1780
1781                                    OFFSET-TO-FAILURE MEASUREMENTS:
1782                                    TRACK  CYLN  SECT  +OFST  -OFST
1783                                                      (UIN)  (UIN)
1784                                      0     0     4   NONE   NONE
1785                                      0     0    20   NONE   NONE
1786                                      0   631     4   NONE   NONE
1787                                      0   631    20   NONE   NONE
1788                                      1     0     4   NONE   NONE
1789                                      1     0    20   NONE   NONE
1790                                      1   631     4   NONE   NONE
1791                                      1   631    20   NONE   NONE
1792                                      2     0     4   NONE   NONE
1793                                      2     0    20   NONE   NONE
1794                                      2   631     4   NONE   NONE
1795                                      2   631    20   NONE   NONE
```

```
1796
1797
1798
1799
1800                                    END PASS #     1
1801                                    TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>
1802                                    *
```

APPENDIX C

SAMPLE MANUAL MODE HEAD ALIGNMENT AID RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING THE HEAD ALIGNMENT AID, PROVIDED IN SUBSYSTEM VERIFICATION PART 2 (ADDRESS 224 START). MANUAL MODE WAS REQUESTED, AND THE OPERATOR SELECTED DRIVE 0 FOR ALIGNMENT. IN THIS CASE, THE OPERATOR SELECTED HEADS 1,0, AND 2, IN THAT ORDER, AND THEN REQUESTED RANDOM SEEK EXERCISES TO BE RUN ON THE DRIVE (AFTER REMOVAL OF THE HEAD ALIGNMENT TOOL). THE EXERCISES WOULD HAVE NORMALLY RUN FOR 5 MINUTES BUT THEY WERE PREMATURELY TERMINATED BY THE OPERATOR, BY TYPING (↑Z), IN THIS PARTICULAR RUN. FINALLY, VERIFY MODE WAS REQUESTED, AND HEAD 2 WAS SELECTED BY THE OPERATOR.

```
DZR6N-C - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2


*** RK06 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A)?
M


* MANUAL SELECT MODE *

ENTER DRIVE NO. (0-7):
0
DRIVE SER. NO.  8

ALIGN, VERIFY, OR EXERCISE (A, V, OR E)?
A


* MANUAL SELECT ALIGNMENT *

ENTER HEAD NO. (0-2):
1
TYPE <R> WHEN READY:
R
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED

ENTER HEAD NO. (0-2):
```

```
1859
1860
1861
1862                                    0
1863                                    TYPE <R> WHEN READY:
1864                                    R
1865                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1866                                    HEAD 0 SELECTED
1867
1868                                    ENTER HEAD NO. (0-2):
1869                                    2
1870                                    TYPE <R> WHEN READY:
1871                                    R
1872                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1873                                    HEAD 2 SELECTED
1874
1875                                    ENTER HEAD NO. (0-2):
1876                                    ↑Z
1877
1878                                    ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1879                                    E
1880                                    TYPE <R> WHEN READY:
1881                                    R
1882
1883
1884                                    *RANDOM SEEK EXERCISES IN PROGRESS ON DRIVE 0 *
1885                                    ↑Z
1886
1887                                    ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1888                                    V
1889
1890
1891                                    * MANUAL SELECT VERIFY *
1892
1893                                    ENTER HEAD NO. (0-2) :
1894                                    2
1895                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1896                                    HEAD 2 SELECTED
1897
1898                                    ENTER HEAD NO. (0-2) :
1899                                    ↑Z
1900
1901                                    ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
```

APPENDIX D

SAMPLE AUTO MODE HEAD ALIGNMENT AID RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING THE HEAD ALIGNMENT AID, PROVIDED IN SUBSYSTEM VERIFICATION PART 2 (ADDRESS 224 START). AUTO MODE WAS REQUESTED, AND THE OPERATOR SELECTED DRIVE 0 FOR ALIGNMENT. IN THIS CASE, HEADS 0,1,2, AND 0 (AGAIN) WERE SELECTED FOR ALIGNMENT (IN AUTO MODE, THE HEADS ARE ALWAYS SELECTED IN THE ORDER 0,1,2,0,1,2, ETC.). BETWEEN HEAD SELECTIONS, THE HEADS WERE UNLOADED TO ALLOW REMOVAL OR INSTALLATION OF THE HEAD ALIGNMENT TOOL. NOTE THAT ALL DRIVE AND HEAD SELECTION (IN AUTO MODE) IS DONE BY THE OPERATOR, AT THE DRIVES, BY OPERATION OF PORT SELECT SWITCHES. AFTER ALIGNING HEADS, THE OPERATOR REQUESTED RANDOM SEEK EXCERCISES TO BE RUN (ON ALL DRIVES, IN THIS CASE DRIVE 0). SHORT EXERCISES WERE REQUESTED, WHICH RUN FOR 1 MINUTE PER DRIVE. FINALLY, AUTO SELECT VERIFY WAS REQUESTED, AND HEADS 0,1, AND 2 WERE SELECTED SEQUENTIALLY.

```
DZR6N-C - RK611/RK06 SUBSYSTEM VERIFICATION : PART 2


*** RK06 HEAD ALIGNMENT AID ***
FOR HELP TYPE H, ELSE <CR>

MANUAL OR AUTO MODE (M OR A):
A


* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
A


* AUTO SELECT ALIGNMENT *

DRIVE 0 SELECTED

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED
```

```
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
↑Z

* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
E

* AUTO SELECT EXERCISES *

SHORT OR LONG (S OR L) ?
S
EXERCISING DRIVE 0

* AUTO SELECT MODE *

ALIGN, VERIFY, OR EXERCISE (A, V, OR E)?
V

* AUTO SELECT VERIFY *

DRIVE 0 SELECTED

HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 0 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 1 SELECTED
HEADS POSITIONED AT CYLINDER 365 (OCT)
HEAD 2 SELECTED
↑Z

ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?


@
```

```
                        PART=1
                        :*** IF "PART" IS DEFINED = 0, PART 1 WILL BE ASSEMBLED. ***
                        :*** IF "PART" IS DEFINED NOT = 0, PART 2 WILL BE ASSEMBLED. ***


                                .NLIST  MC,MD,CND
                                .LIST   ME
                                .ENABL  ABS,AMA
000000  167000                $SWR=   167000
                        ;;*****************************************************************
                        .SBTTL  STARTING ADDRESSES
                        ;*
                        ;*      200     DEFAULT PARAMETERS FOR TESTS 1-6
                        ;*      204     SELECT PARAMETERS FOR TESTS 1-6
                        ;*      224     HEAD ALIGNMENT AID
                        ;;*****************************************************************

000000                  $TN=1
                        .TITLE  MD-11-DZR6N-C - RK611/RK06 SUBSYS. VERIF. : PART 2
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DAVE HOFFMAN
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            REPORT DESCRIPTION ONLY, ON ERRORS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            APPLY RANDOM STALL BETWEEN OPERATIONS
                        ;*         6            REPORT 1 ERROR PER TRANSFER IN TESTS 2-4
                        ;*         1            INHIBIT WRITES IN TEST 1
                        ;*         0            REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 2-4
                        .SBTTL  CONTROL SWITCH SETTINGS (PARAMETER CS)
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        05            DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
                        ;*        04            TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
                        ;*        01            INHIBIT OFFSET REPORTS IN TEST 1

                        .SBTTL  BASIC DEFINITIONS

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

```
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
```

```
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

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

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

                                ;*KT11 VECTOR ADDRESS

                        000250  MMVEC=  250

                                ;*KT11 STATUS REGISTER ADDRESSES
```

```
2170
2171            177572                  SR0=    177572
2172            177574                  SR1=    177574
2173            177576                  SR2=    177576
2174            172516                  SR3=    172516
2175
2176                                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
2177
2178            172300                  KIPDR0= 172300
2179            172302                  KIPDR1= 172302
2180            172304                  KIPDR2= 172304
2181            172306                  KIPDR3= 172306
2182            172310                  KIPDR4= 172310
2183            172312                  KIPDR5= 172312
2184            172314                  KIPDR6= 172314
2185            172316                  KIPDR7= 172316
2186
2187                                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
2188
2189            172340                  KIPAR0= 172340
2190            172342                  KIPAR1= 172342
2191            172344                  KIPAR2= 172344
2192            172346                  KIPAR3= 172346
2193            172350                  KIPAR4= 172350
2194            172352                  KIPAR5= 172352
2195            172354                  KIPAR6= 172354
2196            172356                  KIPAR7= 172356
2197
2198            170200                  MAPLO0=170200
2199            170202                  MAPHO0=170202
2200            172100                  MEMCSR=172100                   ;MEMORY CSR REG START ADRS
2201            177740                  LOERAD=177740                   ;11/70 MEM LO ERROR ADRS REG
2202            177742                  HIERAD=177742                   ;11/70 MEM HI ERROR ADRS REG
2203            177744                  MEMSYS=177744                   ;11/70 MEM SYSTEM REG
2204                                    .SBTTL  TRAP CATCHER
2205
2206            000000                          .=0
2207                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
2208                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
2209                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
2210            000174                          .=174
2211  000174   000000                  DISPREG: .WORD  0                ;;SOFTWARE DISPLAY REGISTER
2212  000176   000000                  SWREG:   .WORD  0                ;;SOFTWARE SWITCH REGISTER
2213                                    .SBTTL  STARTING ADDRESS(ES)
2214  000200   000137  013454                  JMP     @#DFSTRT         ;;JUMP TO STARTING ADDRESS OF PROGRAM
2215            000170                        .=170
2216  000170   000000                  KILLDR: .WORD   0                ;IF NOT = 0, IT IS OK TO OVERLAY LOADER
2217            000204                        .=204
2218  000204   000137  013510                  JMP     @#PSTART
2219            000224                        .=224
2220  000224   000137  023634                  JMP     @#ASTART
2221  000230   000700                  ..LOW:  .WORD   700
2222  000232   001100                  ..HIGH: .WORD   1100
2223                                    .SBTTL  ACT11 HOOKS
2224
2225                                    ;;*****************************************************************
```

```
                        ;HOOKS REQUIRED BY ACT11
000234                          $SVPC=.                 ;SAVE PC
000046                          .=46
000046  023600                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                          .=52
000052  140000                  .WORD   140000          ;;2)SET LOC.52 TO 140000
000234                          .=$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  001320          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  001130          $TSTM:  .WORD   1130    ;;RUN TIM OF LONGEST TEST
001006  003410          $PASTM: .WORD   3410    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  003410          $UNITM: .WORD   3410    ;;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)
```

```
 2315  001224  000000                $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
 2316  001226  000000                $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
 2317  001230  000000                $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
 2318  001232  000000                $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
 2319  001234  000000                $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
 2320  001236  000000                $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
 2321  001240  000000                $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
 2322  001242  000000                $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
 2323  001244  000000                $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
 2324  001246  000000                $REG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
 2325  001250  000000                $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
 2326  001252  000000                $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
 2327  001254  000000                $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
 2328  001256  000000                $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
 2329  001260  000000                $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
 2330  001262  000000                $TMP0:  .WORD   0               ;;USER DEFINED
 2331  001264  000000                $TMP1:  .WORD   0               ;;USER DEFINED
 2332  001266  000000                $TMP2:  .WORD   0               ;;USER DEFINED
 2333  001270  000000                $TMP3:  .WORD   0               ;;USER DEFINED
 2334  001272  000000                $TMP4:  .WORD   0               ;;USER DEFINED
 2335  001274  000000                $TMP5:  .WORD   0               ;;USER DEFINED
 2336  001276  000000                $TMP6:  .WORD   0               ;;USER DEFINED
 2337  001300  000000                $TMP7:  .WORD   0               ;;USER DEFINED
 2338  001302  000000                $TMP10: .WORD   0               ;;USER DEFINED
 2339  001304  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
 2340  001306  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 2341  001310  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
 2342  001314     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
 2343  001315     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 2344  001316  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
 2345                                ;;*****************************************************************
 2346                                .SBTTL  APT MAILBOX-ETABLE
 2347
 2348                                ;;*****************************************************************
 2349                                .EVEN
 2350  001320                        $MAIL:                          ;;APT MAILBOX
 2351  001320  000000                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
 2352  001322  000000                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
 2353  001324  000000                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
 2354  001326  000000                $PASS:  .WORD   APASS           ;;PASS COUNT
 2355  001330  000000                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
 2356  001332  000000                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
 2357  001334  000000                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
 2358  001336  000000                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
 2359  001340                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
 2360  001340     000                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
 2361  001341     000                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
 2362  001342  000000                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
 2363  001344  000000                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
 2364  001346  000000                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
 2365                                :*                              BITS 15-11=CPU TYPE
 2366                                :*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
 2367                                :*                                      11/70=06,PDQ=07,Q=10
 2368                                :*                              BIT 10=REAL TIME CLOCK
 2369                                :*                              BIT  9=FLOATING POINT PROCESSOR
 2370                                :*                              BIT  8=MEMORY MANAGEMENT
```

```
2371  001350  000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
2372  001351  000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
2373                                                      MEM.TYPE BYTE   --  (HIGH BYTE)
2374                              :*                              900 NSEC CORE=001
2375                              :*                              300 NSEC BIPOLAR=002
2376                              :*                              500 NSEC MOS=003
2377  001352  000000             $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
2378                              :*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
2379  001354  000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
2380  001355  000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
2381  001356  000000             $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
2382  001360  000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
2383  001361  000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
2384  001362  000000             $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
2385  001364  000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
2386  001365  000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
2387  001366  000000             $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
2388  001370  120210             $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
2389  001372  000000             $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
2390  001374  177440             $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
2391  001376  000377             $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
2392  001400                     $ETEND:
2393                              .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


001400                          $ERRTB:

                                        ;ERROR 1                ;UNIBUS PARITY ERROR
001400  056711                          EM1
001402  062004                          DH100
001404  063570                          DT100
001406  063704                          DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
001410  056735                          EM2
001412  062004                          DH100
001414  063570                          DT100
001416  063704                          DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
001420  056761                          EM3
001422  062004                          DH100
001424  063570                          DT100
001426  063704                          DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
001430  057004                          EM4
001432  062004                          DH100
001434  063570                          DT100
001436  063704                          DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
001440  057025                          EM5
001442  062004                          DH100
001444  063570                          DT100
001446  063734                          DF02

                                        ;ERROR 6
001450  057044                          EM6                     ;D TO C PARITY ERROR
001452  062004                          DH100
001454  063570                          DT100
001456  063770                          DF03

                                        ;ERROR 7
001460  057070                          EM7                     ;DRIVE DETECTED PARITY ERROR
001462  062004                          DH100
001464  063570                          DT100
```

```
2450  001466  063770                    DF03
2451
2452                                    ;ERROR 10
2453  001470  057124                    EM10                ;AC LOW
2454  001472  062004                    DH100
2455  001474  063570                    DT100
2456  001476  063734                    DF02
2457
2458                                    ;ERROR 11
2459  001500  057133                    EM11                ;SPEED LOSS
2460  001502  062004                    DH100
2461  001504  063570                    DT100
2462  001506  063734                    DF02
2463
2464                                    ;ERROR 12
2465  001510  057146                    EM12                ;ILLEGAL FUNCTION
2466  001512  062004                    DH100
2467  001514  063570                    DT100
2468  001516  063734                    DF02
2469
2470                                    ;ERROR 13
2471  001520  057167                    EM13                ;PROGRAMMING ERROR
2472  001522  062004                    DH100
2473  001524  063570                    DT100
2474  001526  063704                    DF01
2475
2476                                    ;ERROR 14           ;NON-EXISTANT FUNCTION
2477  001530  057211                    EM14
2478  001532  062004                    DH100
2479  001534  063570                    DT100
2480  001536  063734                    DF02
2481
2482                                    ;ERROR 15
2483  001540  057237                    EM15                ;DRIVE TYPE ERROR
2484  001542  062004                    DH100
2485  001544  063570                    DT100
2486  001546  063734                    DF02
2487
2488                                    ;ERROR 16
2489  001550  057260                    EM16                ;FORMAT ERROR
2490  001552  062004                    DH100
2491  001554  063570                    DT100
2492  001556  063734                    DF02
2493
2494                                    ;ERROR 17
2495  001560  057275                    EM17                :WRITE LOCK ERROR
2496  001562  062004                    DH100
2497  001564  063570                    DT100
2498  001566  063734                    DF02
2499
2500                                    ;ERROR 20
2501  001570  057316                    EM20                ;DRIVE UNSAFE
2502  001572  062004                    DH100
2503  001574  063570                    DT100
2504  001576  063734                    DF02
2505
```

```
2506
2507  001600  057333          ;ERROR 21
2508  001602  062004          EM21                    ;SEEK INCOMPLETE
2509  001604  063570          DH100
2510  001606  063734          DT100
2511                          DF02
2512
2513  001610  057353          ;ERROR 22
2514  001612  062004          EM22                    ;CYLINDER OVERFLOW
2515  001614  063570          DH100
2516  001616  063734          DT100
2517                          DF02
2518
2519  001620  057375          ;ERROR 23
2520  001622  062004          EM23                    ;ILLEGAL CYLINDER
2521  001624  063570          DH100
2522  001626  063734          DT100
2523                          DF02
2524
2525  001630  057426          ;ERROR 24
2526  001632  062004          EM24                    ;DRIVE OFF TRACK
2527  001634  063570          DH100
2528  001636  063734          DT100
2529                          DF02
2530
2531  001640  057446          ;ERROR 25
2532  001642  062004          EM25                    ;DRIVE TIMING ERROR
2533  001644  063570          DH100
2534  001646  063734          DT100
2535                          DF02
2536
2537  001650  057471          ;ERROR 26
2538  001652  062004          EM26                    ;DATA LATE
2539  001654  063570          DH100
2540  001656  063734          DT100
2541                          DF02
2542
2543  001660  057503          ;ERROR 27
2544  001662  062004          EM27                    ;CONTROLLER TIMEOUT
2545  001664  063570          DH100
2546  001666  063734          DT100
2547                          DF02
2548
2549  001670  057526          ;ERROR 30               ;OPERATION INCOMPLETE
2550  001672  062004          EM30
2551  001674  063570          DH100
2552  001676  064044          DT100
2553                          DF05
2554
2555  001700  057553          ;ERROR 31
2556  001702  062004          EM31                    ;HEADER VRC ERROR
2557  001704  063570          DH100
2558  001706  064044          DT100
2559                          DF05
2560
2561  001710  057574          ;ERROR 32
                              EM32                    ;DATA CHECK ERROR
```

```
2562  001712  062004                  DH100
2563  001714  063570                  DT100
2564  001716  064100                  DF07
2565
2566                                  ;ERROR 33
2567  001720  057615                  EM33                ;WRITE CHECK ERROR
2568  001722  062004                  DH100
2569  001724  063570                  DT100
2570  001726  064134                  DF10
2571
2572                                  ;ERROR 34
2573  001730  057637                  EM34                ;DATA MISCOMPARE(S)
2574  001732  062004                  DH100
2575  001734  063570                  DT100
2576  001736  064030                  DF04
2577
2578                                  ;ERROR 35
2579  001740  057657                  EM35                ;NO DRIVE RESPONSE-UFE & NXD
2580  001742  062004                  DH100
2581  001744  063570                  DT100
2582  001746  063704                  DF01
2583
2584                                  ;ERROR 36
2585  001750  057713                  EM36                ;DRIVE ERROR WILL NOT CLEAR
2586  001752  000000                  0
2587  001754  000000                  0
2588  001756  000000                  0
2589
2590                                  ;ERROR 37           ;DRIVE STATUS CHANGE WILL NOT CLEAR
2591  001760  057746                  EM37
2592  001762  000000                  0
2593  001764  000000                  0
2594  001766  000000                  0
2595
2596                                  ;ERROR 40
2597  001770  060011                  EM40                ;ATTENTION BUT NO STATUS CHANGE OR FAULT
2598  001772  062004                  DH100
2599  001774  063570                  DT100
2600  001776  063734                  DF02
2601
2602                                  ;ERROR 41
2603  002000  060055                  EM41                ;ATTENTION BUT DRIVE NOT AVAILABLE
2604  002002  062004                  DH100
2605  002004  063570                  DT100
2606  002006  063734                  DF02
2607
2608                                  ;ERROR 42
2609  002010  060113                  EM42                ;ATTENTION WHEN NOT EXPECTED
2610  002012  062004                  DH100
2611  002014  063570                  DT100
2612  002016  063734                  DF02
2613
2614                                  ;ERROR 43           ;ERROR WHILE GATHERING DRIVE STATUS
2615  002020  060143                  EM43
2616  002022  062004                  DH100
2617  002024  063570                  DT100
```

```
002026  064200          DF12

                        ;ERROR 44
002030  060420          EM63                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
002032  062004          DH100
002034  063570          DT100
002036  064200          DF12

                        ;ERROR 45
002040  060465          EM64                    ;NO ATTENTION IN ATTENTION SUMMARY REG
002042  062004          DH100
002044  063570          DT100
002046  064200          DF12

                        ;ERROR 46
002050  060530          EM65                    ;UNSOLICITED ATTENTION
002052  062004          DH100
002054  063570          DT100
002056  064200          DF12

                        ;ERROR 47
002060  060556          EM66                    ;UNEXPECTED DATA TYPE ERROR
002062  062004          DH100
002064  063570          DT100
002066  064200          DF12

                        ;ERROR 50
002070  060611          EM67                    ;ATTENTION DID NOT RESET WITH CLEAR
002072  062004          DH100
002074  063570          DT100
002076  064200          DF12

                        ;ERROR 51
002100  060650          EM70                    ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
002102  062004          DH100
002104  063570          DT100
002106  064200          DF12

                        ;ERROR 52
002110  060200          EM52                    ;MULTIPLE DRIVE SELECT
002112  062004          DH100
002114  063570          DT100
002116  064200          DF12

                        ;ERROR 53
002120  060226          EM53                    ;ABREVIATED HCE ERROR
002122  062004          DH100
002124  063570          DT100
002126  064230          DF13

                        ;ERROR 54
002130  057526          EM30                    ;OPERATION INCOMPLETE ERROR
002132  062004          DH100
002134  063570          DT100
002136  064260          DF14
```

```
002674
002676  002140  057553          ;ERROR 55
002700  002142  062004          EM31                    ;ABBEVIATED HVRC ERROR
002702  002144  063570          DH100
002704  002146  064230          DT100
002706                          DF13
002710
002712                          ;ERROR 56
002714  002150  060253          EM56                    ;2 TIMEOUT ERROR
002716  002152  062004          DH100
002720  002154  063570          DT100
002722  002156  064310          DF15
002724
002726                          ;ERROR 57               ;2ND LEVEL IN SUBSYSTEM TIMEOUT
002730  002160  060253          EM56
002732  002162  062004          DH100
002734  002164  063570          DT100
002736  002166  064354          DF16
002740
002742                          ;ERROR 60
002744  002170  060272          EM60                    ;ERROR IN RECAL FOR RECOVERY
002746  002172  000000          0
002750  002174  000000          0
002752  002176  000000          0
002754
002756                          ;ERROR 61
002760  002200  060326          EM61                    ;ABORT MESSAGE
002762  002202  000000          0
002764  002204  000000          0
002766  002206  000000          0
002770
002772                          ;ERROR 62
002774  002210  060374          EM62                    ;CYLINDER MISCOMPARE
002776  002212  062004          DH100
003000  002214  063570          DT100
003002  002216  064420          DF17
003004
003006                          ;ERROR 63               ;DATA ERROR WORDS
003010  002220  000000          0
003012  002222  000000          0
003014  002224  063662          DT602
003016  002226  064530          DF25
003020
003022                          ;ERROR 64
003024  002230  060420          EM63                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
003026  002232  062004          DH100
003030  002234  063570          DT100
003032  002236  063734          DF02
003034
003036                          ;ERROR 65
003040  002240  060465          EM64                    ;NO ATTENTION IN ATTENTION SUMMARY REG
003042  002242  062004          DH100
003044  002244  063570          DT100
003046  002246  063734          DF02
003050
003052                          ;ERROR 66
003054  002250  060530          EM65                    ;UNSOLICITED ATTENTION
```

```
002730  002252  062004                  DH100
002731  002254  063570                  DT100
002732  002256  063734                  DF02
002733
002734                                  ;ERROR 67
002735  002260  060556                  EM66            ;UNEXPECTED DATA TYPE ERROR
002736  002262  062004                  DH100
002737  002264  063570                  DT100
002738  002266  063734                  DF02
002739
002740                                  ;ERROR 70
002741  002270  060611                  EM67            ;ATTENTION DID NOT RESET WITH CLEAR
002742  002272  062004                  DH100
002743  002274  063570                  DT100
002744  002276  063734                  DF02
002745
002746                                  ;ERROR 71
002747  002300  060650                  EM70            ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
002748  002302  062004                  DH100
002749  002304  063570                  DT100
002750  002306  063734                  DF02
002751
002752                                  ;ERROR 72
002753  002310  060717                  EM71            ;DATA LATE WHEN UNLOADING HEADER
002754  002312  062004                  DH100
002755  002314  063570                  DT100
002756  002316  063734                  DF02
002757
002758                                  ;ERROR 73
002759  002320  060757                  EM72            ;CONTROLLER ERROR DURING DRIVER SERVICE
002760  002322  062004                  DH100
002761  002324  063570                  DT100
002762  002326  063734                  DF02
002763
002764                                  ;ERROR 74
002765  002330  061026                  EM73            ;DRIVE DETECTED PARITY ERROR
002766  002332  062004                  DH100
002767  002334  063570                  DT100
002768  002336  063734                  DF02
002769
002770                                  ;ERROR 75
002771  002340  061062                  EM74            ;UNDEFINED ERROR
002772  002342  062004                  DH100
002773  002344  063570                  DT100
002774  002346  063734                  DF02
002775
002776                                  ;ERROR 76
002777  002350  061102                  EM75            ;MARKING SECTOR BAD MESSAGE
002778  002352  000000                  0
002779  002354  000000                  0
002780  002356  000000                  0
002781
002782                                  ;ERROR 77
002783  002360  061132                  EM76            ;BAD DATA VERIFICATION WITH READ
002784  002362  062771                  DH605
002785  002364  063654                  DT601
```

```
2786  002366  064454                    DF21
2787
2788                                    ;ERROR 100
2789  002370  061214                    EM77            ;RETRY SUCCESFUL MESSAGE
2790  002372  000000                    0
2791  002374  063654                    DT601
2792  002376  064514                    DF23
2793
2794                                    ;ERROR 101
2795  002400  061214                    EM77            ;ANOTHER RETRY SUCCESSFUL MESSAGE
2796  002402  000000                    0
2797  002404  063654                    DT601
2798  002406  064514                    DF23
2799
2800                                    ;ERROR 102
2801  002410  061235                    EM100           ;RETRY UNSUCCESSFUL MESSAGE
2802  002412  000000                    0
2803  002414  063654                    DT601
2804  002416  064514                    DF23
2805
2806                                    ;ERROR 103
2807  002420  061260                    EM101           ;NO VALID HEADERS IN TRACK JUST READ
2808  002422  062753                    DH6042
2809  002424  063654                    DT601
2810  002426  064524                    DF24
2811
2812                                    ;ERROR 104
2813  002430  061336                    EM102           ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2814  002432  062004                    DH100
2815  002434  063570                    DT100
2816  002436  063734                    DF02
2817
2818                                    ;ERROR  105
2819  002440  061410                    EM103           ;TIMED-OUT ON READ HEADER
2820  002442  062004                    DH100
2921  002444  063570                    DT100
2822  002446  063770                    DF03
2823
2824                                    ;ERROR  106
2825  002450  061436                    EM104           ;TIMED-OUT ON SEEK
2826  002452  062004                    DH100
2827  002454  063570                    DT100
2828  002456  063770                    DF03
2829
2830                                    ;ERROR  107
2831  002460  061460                    EM105           ;DRIVE SIEZED BY OTHER PORT
2832  002462  062004                    DH100
2833  002464  063570                    DT100
2834  002466  063734                    DF02
2835
2836                                    ;ERROR  110
2837  002470  061513                    EM106           ;"DATA MISCMPR WHILE BAI SET"
2838  002472  062004                    DH100
2839  002474  063570                    DT100
2840  002476  064544                    DF27
2841
```

```
2842                                    ;ERROR  111
2843  002500  061546                   EM107                   ;"NO MEM WHEN EXPECTED"
2844  002502  000000                   0
2845  002504  000000                   0
2846  002506  000000                   0
2847
2848                                    ;ERROR  112
2849  002510  061613                   EM110                   ;"INTRPT WHEN CNTRLR NOT READY"
2850  002512  062004                   DH100
2851  002514  063570                   DT100
2852  002516  063704                   DF01
2853
2854                                    ;ERROR  113
2855  002520  061646                   EM111                   ;"NO ATT'N ON SEEK"
2856  002522  062004                   DH100
2857  002524  063570                   DT100
2858  002526  063734                   DF02
2859
2860                                    ;ERROR  114
2861  002530  061667                   EM112                   ;DRIVE'S CYLINDER INCORRECT
2862  002532  063200                   DH702
2863  002534  063630                   DT202
2864  002536  064534                   DF26
2865
2866                                    ;ERROR  115
2867  002540  000000                   0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2868  002542  000000                   0
2869  002544  063654                   DT601
2870  002546  064160                   DF11
2871
2872                                    ;ERROR  116
2873  002550  057637                   EM34                    ;DATA MISCOMPARE (11/70)
2874  002552  062315                   DH103
2875  002554  063700                   DT103
2876  002556  064444                   DF20
2877
2878                                    ;ERROR  117
2879  002560  000000                   0                       ;PART OF DATA MISCOMPARE
2880  002562  000000                   0
2881  002564  063570                   DT100
2882  002566  064464                   DF22
2883
2884                                    ;ERROR  120
2885  002570  061722                   EM113                   ;ABORT- CAN'T READ BSF
2886  002572  062004                   DH100
2887  002574  063570                   DT100
2888  002576  063704                   DF01
2889
2890                                    ;ERROR  121
2891  002600  061750                   EM114                   ;KT11 FAILURE
2892  002602  062004                   DH100
2893  002604  063570                   DT100
2894  002606  064570                   DF30
2895
2896                                    ;ERROR  122
2897  002610  061765                   EM115                   ;MEM PARITY ERROR
```

```
2898  002612  062004                    DH100
2899  002614  063570                    DT100
2900  002616  064614                    DF31
2901
2902
2903
2904
2905
2906
```

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

```
;RKER (ERROR REGISTER)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; DCK ! UNS  ! OPI  ! DTE  ! WLE  ! IDAE ! COE  ! HVRC !
;                    READ ONLY                        !
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
; BSE ! ECH  ! DTYPE! FMTE ! DRPAR! ILF  ! SKI  ! ILC  !
;                    READ ONLY                        !
;-----------------------------------------------------

;RKDS (DRIVE STATUS)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;SVAL ! DSC  ! PIP  ! SPON ! WRL  !  UN  !  UN  ! DTP  !
;                    READ ONLY                        !
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
;DRDY !  VV  ! DROT ! SPLS ! ACLO ! OFSET!  UN  ! DRA  !
;                    READ ONLY                        !
;-----------------------------------------------------

;RKMR1 (MAINTENANCE REG 1)
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
;RD   ! WRT  ! ECCW ! PCD  ! PCA  ! MEWD ! MERD ! MC_K !
;GATE ! GATE !      !      !      !      !      !       !
;                READ ONLY                 ! READ/WRITE !
;-----------------------------------------------------

;  7     6      5      4      3      2      1      0
;-----------------------------------------------------
;MIND ! MSP  ! DMD  ! PAT  ! MS3  ! MS2  ! MS1  ! MS0  !
;                    READ/WRITE                       !
;-----------------------------------------------------
```

```
3051                                    ;RKECC/PAT
3052                                    ; 15     14     13     12     11     10     9      8
3053                                    ;-------------------------------------------------------
3054                                    ; UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
3055                                    ;                    READ ONLY                          !
3056                                    ;-------------------------------------------------------
3057                                    ;
3058                                    ;  7      6      5      4      3      2      1      0
3059                                    ;-------------------------------------------------------
3060                                    ;EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
3061                                    ;                    READ ONLY                          !
3062                                    ;-------------------------------------------------------
3063                                    ;
3064                                    ;RKECC/POS
3065                                    ; 15     14     13     12     11     10     9      8
3066                                    ;-------------------------------------------------------
3067                                    ; UN  !  UN  !  UN  ! EPO12! EPO11! EPO10! EPO9 ! EPO8 !
3068                                    ;                    READ ONLY                          !
3069                                    ;-------------------------------------------------------
3070                                    ;
3071                                    ;  7      6      5      4      3      2      1      0
3072                                    ;-------------------------------------------------------
3073                                    ;EPO7 ! EPO6 ! EPO5 ! EPO4 ! EPO3 ! EPO2 ! EPO1 ! EPO0 !
3074                                    ;                    READ ONLY                          !
3075                                    ;-------------------------------------------------------
3076
3077                                    ;DRIVE STATUS INFORMATION
3078
3079                                    ;LINE A MESSAGE 00
3080                                    ; 15     14     13     12     11     10     9      8
3081                                    ;-------------------------------------------------------
3082                                    ; PAR ! DSC  ! PIP  ! SPON ! WRLK ! OFFON! FMT  ! DRTP !
3083                                    ;-------------------------------------------------------
3084                                    ;
3085                                    ;  7      6      5      4      3      2      1      0
3086                                    ;-------------------------------------------------------
3087                                    ;DRDY !  VV  ! DRA  !  NU  !  NU  ! DRIVE SELECT CODE  !
3088                                    ;-------------------------------------------------------
```

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

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

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

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

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

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

```
                                ;LINE A MESSAGE 10
                                ; 15     14     13     12     11     10     9      8
                                ;-------------------------------------------------------
                                ; PAR !  NU  !  NU  ! CYLINDER DIFF/OFFSET VALUE       !
                                ;-------------------------------------------------------

                                ;  7      6      5      4      3      2      1      0
                                ;-------------------------------------------------------
                                ;CYLINDER DIFF/OFFSET VALUE!  NU  !  DRIVE SELECT CODE !
                                ;-------------------------------------------------------

                                ;LINE B MESSAGE 10
                                ; 15     14     13     12     11     10     9      8
                                ;-------------------------------------------------------
                                ; PAR ! ALIGN!  NU  !    CYLINDER ADDRESS              !
                                ;     ! SIGN !                                        !
                                ;-------------------------------------------------------

                                ;  7      6      5      4      3      2      1      0
                                ;-------------------------------------------------------
                                ;    CYLINDER ADDRESS      ! UN  ! UN  !   1  !   0  !
                                ;-------------------------------------------------------

                                ;LINE A MESSAGE 11
                                ; 15     14     13     12     11     10     9      8
                                ;-------------------------------------------------------
                                ; PAR !     DRIVE SERIAL NUMBER                        !
                                ;-------------------------------------------------------

                                ;  7      6      5      4      3      2      1      0
                                ;-------------------------------------------------------
                                ; DRIVE SERIAL NUMBER             !  DRIVE SELECT CODE !
                                ;-------------------------------------------------------

                                ;LINE B MESSAGE 11
                                ; 15     14     13     12     11     10     9      8
                                ;-------------------------------------------------------
                                ; PAR !  NU  !     NU      !   DECODED HEAD     ! SECTOR!
                                ;     !      !             !    ADDRESS         ! COUNT !
                                ;-------------------------------------------------------

                                ;  7      6      5      4      3      2      1      0
                                ;-------------------------------------------------------
                                ;   SECTOR COUNT           ! UN  ! UN  !   1  !   1  !
                                ;-------------------------------------------------------


                                .SBTTL  RK06 CONTROLLER REGISTER DEFINITION

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

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

                        .SBTTL  DRIVE COMMANDS

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

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

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

                        ;       DRIVER ISSUED SERVICE COMMANDS

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

                        .SBTTL  CONTROL AND STATUS REGISTER 1 BITS

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

```
                        ;CONTROL AND STATUS REGISTER 1 BITS

000040          SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
000100          DI=     BIT14                   ;DRIVE INTERRUPT
000200          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)
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
```

```
                        STATUS REGISTER BIT DEFINITION

000000                  DRA=    BIT0            ;DRIVE AVAILABLE (CONTROLLER IS SET IF
                                                ;  THIS BIT IS RESET)
000004                  OFST=   BIT2            ;DRIVE OFFSET
000010                  ACLO=   BIT3            ;AC LOW
000020                  SPDLSS= BIT4            ;SPEED LOSS
000020                  DCLO=   BIT4            ;DC LOW
000040                  DROT=   BIT5            ;DRIVE OFF TRACK
000100                  VV=     BIT6            ;VOLUME VALID
000200                  DRY=    BIT7            ;DRIVE READY
000200                  DROY=   BIT7            ;DRIVE READY
000400                  DCT=    BIT8            ;DRIVE TYPE (0=RK06)
004000                  WRL=    BIT11           ;WRITE LOCK
020000                  PIP=    BIT13           ;POSITIONING IN PROGRESS
040000                  DSC=    BIT14           ;DRIVE STATUS CHANGE
100000                  SVAL=   BIT15           ;STATUS VALID

                        .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION

000017                  MESMSK= 17              ;MESSAGE MASK

000020                  PAT=    BIT4            ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
000040                  DMD=    BIT5            ;DIAGNOSTIC MODE
000100                  MSP=    BIT6            ;MAINTENANCE SECTOR PULSE
000200                  MIND=   BIT7            ;MAINTENANCE INDEX
000400                  MCLK=   BIT8            ;MAINTENANCE CLOCK
001000                  MERD=   BIT9            ;MAINTENANCE ENCODED READ DATA
002000                  MEWD=   BIT10           ;MAINTENACNE ENCODED WRITE DATA
004000                  PCA=    BIT11           ;PRECOMPENSATION ADVANCE
010000                  PCD=    BIT12           ;PRECOMPENSATION DELAY
020000                  ECCW=   BIT13           ;ECC WORD IS BEING READ OR WRITTEN
040000                  WRTGAT= BIT14           ;WRITE GATE
100000                  RDGATE= BIT15           ;READ GATE

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A

000040                  S.DRA=  BIT5            ;DRIVE AVAILIABLE
000100                  S.VV=   BIT6            ;VOLUME VALID
000200                  S.DRY=  BIT7            ;DRIVE READY
000400                  S.TYPE= BIT8            ;DRIVE TYPE
001000                  S.FORM= BIT9            ;DRIVE FORMAT
002000                  S.OFF=  BIT10           ;OFFSET
004000                  S.WRL=  BIT11           ;WRITE LOCK
010000                  S.SPIN= BIT12           ;SPINDLE ON
020000                  S.PIP=  BIT13           ;POSITIONING IN PROGRESS
040000                  S.DSC=  BIT14           ;DRIVE STATUS CHANGE

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

000040                  S.ICYL= BIT5            ;ILLEGAL CYLINDER ADDRESS
000100                  S.ACLO= BIT6            ;AC LOW
000200                  S.FLT=  BIT7            ;DRIVE FAULT
000400                  S.ILF=  BIT8            ;ILLEGAL FUNCTION
001000                  S.PAR=  BIT9            ;DRIVE DETECTED DRIVE BUS PARITY ERROR
002000                  S.SKI=  BIT10           ;SEEK INCOMPLETE
004000                  S.WLE=  BIT11           ;WRITE LOCK ERROR
010000                  S.SPLS= BIT12           ;SPEED LOSS
```

```
                        ;DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

000000                  S.DCLO= BIT12                   ;DC LOW
000000                  S.DROT= BIT13                   ;DRIVE OFF TRACK
000000                  S.UNS=  BIT14                   ;DRIVE UNSAFE

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A

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

                        .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B

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

                        .SBTTL  COMMON MASKS

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

```
                        .SBTTL  PARAMETER BLOCK ALLOCATION

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

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

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

000000                  P.DRVN= 0                       ;DRIVE NUMBER
000001                  P.CMND= 1                       ;COMMAND
000002                  P.CYLN= 2                       ;CYLINDER ADDRESS
000004                  P.SECT= 4                       ;SECTOR
000005                  P.TRCK= 5                       ;TRACK
000006                  P.OFST= 6                       ;OFFSET
000007                  P.CSIH= 7                       ;RKCSI 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

002620  000         PARM0:  .BYTE   0                               ;DRIVE NUMBER
002621  000                 .BYTE   0                               ;COMMAND
002622  000000              .WORD   0                               ;CYLINDER ADDRESS
002624  000                 .BYTE   0                               ;SECTOR ADDRESS
002625  000                 .BYTE   0                               ;TRACK ADDRESS
002626  000                 .BYTE   0                               ;OFFSET VALUE
002627  000                 .BYTE   0                               ;BUS ADDRESS (BITS 16 AND 17)
002630  000000              .WORD   0                               ;BUS ADDRESS (BITS 0 - 15)
002632  000000              .WORD   0                               ;WORD COUNT (2'S COMPLEMENT)
002634  000000              .WORD   0                               ;PROGRAM DRIVE STATUS INFORMATION
002636  000000              .WORD   0                               ;COMMAND AND STATUS REGISTER 1
002640  000000              .WORD   0                               ;COMMAND AND STATUS REGISTER 2
002642  000000              .WORD   0                               ;WORD COUNT REGISTER
002644  000000              .WORD   0                               ;BUS ADDRESS REGISTER
002646  000000              .WORD   0                               ;DESIRED TRACK AND SECTOR REGISTER
```

```
003505  002650  000000                  .WORD   0               ;DESIRED CYLINDER REGISTER
003506  002652  000000                  .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
003507  002654  000000                  .WORD   0               ;ERROR REGISTER
003508  002656  000000                  .WORD   0               ;STATUS REGISTER
003509  002660  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
003510  002662  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
003511  002664  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
003512  002666  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
003513  002670  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
003514  002672  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
003515  002674  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
003516  002676  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
003517  002700  000000                  .WORD   0               ;ECC POSITION INFORMATION
003518  002702  000000                  .WORD   0               ;ECC PATTERN INFORMATION
003519
003520                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
003521
003522  002704     000          PARM1:  .BYTE   0               ;DRIVE NUMBER
003523  002705     000                  .BYTE   0               ;COMMAND
003524  002706  000000                  .WORD   0               ;CYLINDER ADDRESS
003525  002710     000                  .BYTE   0               ;SECTOR ADDRESS
003526  002711     000                  .BYTE   0               ;TRACK ADDRESS
003527  002712     000                  .BYTE   0               ;OFFSET VALUE
003528  002713     000                  .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
003529  002714  000000                  .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
003530  002716  000000                  .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
003531  002720  000000                  .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
003532  002722  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 1
003533  002724  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 2
003534  002726  000000                  .WORD   0               ;WORD COUNT REGISTER
003535  002730  000000                  .WORD   0               ;BUS ADDRESS REGISTER
003536  002732  000000                  .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
003537  002734  000000                  .WORD   0               ;DESIRED CYLINDER REGISTER
003538  002736  000000                  .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
003539  002740  000000                  .WORD   0               ;ERROR REGISTER
003540  002742  000000                  .WORD   0               ;STATUS REGISTER
003541  002744  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
003542  002746  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
003543  002750  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
003544  002752  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
003545  002754  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
003546  002756  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
003547  002760  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
003548  002762  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
003549  002764  000000                  .WORD   0               ;ECC POSITION INFORMATION
003550  002766  000000                  .WORD   0               ;ECC PATTERN INFORMATION
003551
003552
003553                          .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
003554
003554  002770  000000          T.CS1:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
003555                                                          ;  REGISTER 1
003556  002772  000000          T.CS2:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
003557                                                          ;  REGISTER 2
003558  002774  000000          T.WCR:  .WORD   0               ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
003559  002776  000000          T.BA:   .WORD   0               ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
003560  003000  000000          T.DA:   .WORD   0               ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
3561  003002  000000          T.DC:   .WORD   0           ;TEMPORARY STORAGE FOR DRIVE CYLINDER
3562  003004  000000          T.ASOF: .WORD   0           ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
3563                                                      ;  AND OFFSET
3564  003006  000000          T.ER:   .WORD   0           ;TEMPORARY STORAGE FOR ERROR REGISTER
3565  003010  000000          T.DS:   .WORD   0           ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3566  003012  000000          T.MR1:  .WORD   0           ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3567  003014  000000          T.MR2:  .WORD   0           ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3568  003016  000000          T.MR3:  .WORD   0           ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3569  003020  000000          T.POS:  .WORD   0           ;TEMPORARY STORAGE FOR ECC POSITION
3570  003022  000000          T.PAT:  .WORD   0           ;TEMPORARY STORAGE FOR ECC PATTERN
3571  003024  000000          T.DB:   .WORD   0           ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
3572
3573                          .SBTTL  DRIVER PARAMERTERS
3574
3575  003026  177440          RKBAS:  .WORD   177440      ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3576  003030  000210          RKVEC:  .WORD   210         ;ADDRESS OF R611 VECTOR
3577  003032  000240          RKPRI:  .WORD   PR5         ;RK611 INTERRUPT PRIORITY
3578  003034  041220          A.NORM: ERRFRE              ;ADDRESS OF NORMAL RETURN FROM DRIVER
3579  003036  042466          A.ABNL: ERRHDL              ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
3580  003040  041762          A.CONT: CONERR              ;ADDRESS OF CONTROLLER ERROR RETURN
3581  003042  000000          E.CONT: .WORD   0           ;CONTROLLER ERROR STATUS
3582                                                      ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
3583                                                      ;  IS INITIATED.  IF A CONTROLLER ERROR
3584                                                      ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
3585                                                      ;  USED:
3586                                                      ;
3587          000001          E.CCLR= BIT0                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
3588          000002          E.NOAT= BIT1                ;NO ATTENTION IN ATTENTION SUMMARY REG
3589          000004          E.UATT= BIT2                ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3590          000010          E.UDAT= BIT3                ;UNEXPECTED DATA TYPE ERROR
3591          000020          E.CLAT= BIT4                ;ATTENTION DID NOT RESET WITH CLEAR
3592          000040          E.SCLR= BIT5                ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
3593                                                      ;  ATTENTION
3594          000100          E.ILLD= BIT6                ;ILLEGAL DRIVER COMMAND
3595          000400          E.DLT=  BIT8                ;DATA LATE WHEN UNLOADING HEADER
3596          001000          E.CERR= BIT9                ;CONTROLLER ERROR DURING DRIVER SERVICING
3597          002000          E.DPAR= BIT10               ;DRIVE DETECTED PARITY ERROR
3598          040000          E.CMTO= BIT14               ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3599          100000          E.MDS=  BIT15               ;MULTIPLE DRIVE SELECT
3600
3601  003044  000000          O.WAIT: .WORD   0           ;PARAMETER BLOCK OF THE DRIVE
3602                                                      ;WAITING FOR COMMAND COMPLETION
3603  003046  000400          W.MTIM: .WORD   400         ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
3604  003050  000400          W.MI_I: .WORD   400         ;16 MILISECOND TIME FOR PROGRAM
3605                                                      ;
3606                                                      ;     CPU     VALUE
3607                                                      ;     ---     -----
3608                                                      ;
3609                                                      ;     11/05   100
3610                                                      ;     11/10
3611                                                      ;     11/20
3612                                                      ;     11/34
3613                                                      ;     11/40
3614                                                      ;     11/45   400
3615                                                      ;     11/50
3616                                                      ;     11/70
```

```
                                DRIVER PARAMETERS

003052  000300          W.SEC:  .WORD   300             ;SECOND COUNT COUNT FOR ALL COMMANDS
                                                        ;  EXCEPT START SPINDLE
003054  003000          W.8SEC: .WORD   3000            ;8 SECOND FOR DRIVE CYCLE DOWN
003056  030000          W.MIN:  .WORD   30000           ;MINUTE TIME FOR START SPINDLE
003060  000000          HDR.AD: .WORD   0               ;ADDRESS USED FOR READ ALL HEADERS
003062  000000          HDR.CT: .WORD   0               ;NUMBER OF HEADERS LEFT TO READ FOR READ
                                                        ;  ALL HEADERS
003064      000         I.ISRL: .BYTE   0               ;INTERRUPT OR RELEASED COMMAND ISSUED
003065      002     004     010 H.HEAD: .BYTE   2,4,10      ;HEAD DECODES
003070      000         W.TIME: .BYTE   0               ;DRIVES BEING WATCH-DOG TIMED

                                .SBTTL  INTERRUPT MASKS

003071      000         INTMSK: .BYTE   0               ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK

                                ;       INTERRUPT MASK TABLE

003072      001         I.DRV:  .BYTE   1               ;INTERRUPT MASK FOR DRIVE 0
003073      002                 .BYTE   2               ;INTERRUPT MASK FOR DRIVE 1
003074      004                 .BYTE   4               ;INTERRUPT MASK FOR DRIVE 2
003075      010                 .BYTE   10              ;INTERRUPT MASK FOR DRIVE 3
003076      020                 .BYTE   20              ;INTERRUPT MASK FOR DRIVE 4
003077      040                 .BYTE   40              ;INTERRUPT MASK FOR DRIVE 5
003100      100                 .BYTE   100             ;INTERRUPT MASK FOR DRIVE 6
003101      200                 .BYTE   200             ;INTERRUPT MASK FOR DRIVE 7

                                .SBTTL  PARAMETER BLOCK TABLE

003102  002620          PBLKT:  PARM0                   ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
                                                        ;DRIVE CALL.  MUST BE LOADED INTO PBLKT

                                .SBTTL  TIME FOR WATCH-DOG TIMER

003104  000000          W.DRV:  .WORD   0               ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
                                .SBTTL  PROGRAM SPECIFIC RESERVED LOCATIONS
003106      000         MDFLAG: .BYTE   0               ;FLAG TO INDIC. DEFLT OR PARAM MODE
003107      000         XXDPCH: .BYTE   0               ;XXDP CHAIN MODE FLAG
003110      000         TSTING: .BYTE   0               ;CURRENTLY RUNNING TESTS IF = 1
003111      000         DERCNT: .BYTE   0               ;DATA ERROR COUNT
003112      000         OPCOMP: .BYTE   0               ;OPERATION COMPLETE FLAG
003113      000         DONE:   .BYTE   0               ;DONE SWITCH
003114      000         TYPFMT: .BYTE   0               ;DRIVE TYPE & FORMAT CONTROL
003115      000         FORMAT: .BYTE   0               ;DRIVE FORMAT IN BIT 4 OF BYTE
003116      000         ERRCNT: .BYTE   0               ;ERROR COUNT
003117      004         ERRLMT: .BYTE   4               ;ERROR LIMIT
003120      000         DRVERS: .BYTE   0               ;ERROR COUNT FOR CURRENT DRIVE
003121      000         OPCONT: .BYTE   0               ;OPERATION CONTROL SWITCHES
003122      000         PCLKF:  .BYTE   0               ;IF BYTE=1, KW11-P CLOCK IS PRESENT
003123      000         DOTIM:  .BYTE   0               ;IF BYTE=1, DO TIMING TESTS
003124      000         XOVLAD: .BYTE   0               ;FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
003125      000         XDPSVD: .BYTE   0               ;FLAG = 1 IF XXDP IS SAVED
003126      000         DULACS: .BYTE   0               ;FLAG=1 IF DUAL ACCESS TEST
003127      000         DRNAFG: .BYTE   0               ;=1 INDICATES DRIVE SIEZED BY OTHER PORT
003130      000         REISSU: .BYTE   0               ;DUAL-ACC FLAG TO RE-ISSUE COMMAND
003131      000         WCEFLG: .BYTE   0               ;WRITE CHECK ERROR FLAG
```

```
                                DLTFLG: .BYTE   0               ;DATA LATE ERROR FLAG
                                NORTRY: .BYTE   0               ;"NO-RETRY" FLAG
                                UBMPRS: .BYTE   0               ;UNIBUS MAP PRESENT IF = 1
                                MEMABT: .BYTE   0               ;SET BYTE = 1 FOR NO ABORT
                                                                ;    ON MEMORY PARITY ERRORS
                                HLPOVL: .BYTE   0               ;HELP FILE OVERLAID INDICATOR
                                NOTYPE: .BYTE   0               ;INDICATES PROGRAM JUST LOADED IF 0
                        000001  WHDSW=BIT0                      ;WRITE HEADER & DATA SWITCH
                        000002  VFHDSW=BIT1                     ;VERIFY HEADERS SWITCH
                        000004  WCDASW=BIT2                     ;WRITE CHECK DATA SWITCH
                        000010  RCDASW=BIT3                     ;READ CHECK DATA SWITCH
                        000020  OREQSW=BIT4                     ;OFFSET REQUIRED SWITCH
                                        .EVEN
003140  177546          LKS:    .WORD   177546          ;KW11-L CLOCK STATUS REGISTER
003142  172540          PKS:    .WORD   172540          ;KW11-P CONTROL AND STATUS REGISTER
003144  172542          PKSB:   .WORD   172542          ;KW11-P COUNT SET BUFFER REGISTER
003146  172544          PKRB:   .WORD   172544          ;KW11-P COUNTER REGISTER
003150  000100          LCVEC:  .WORD   100             ;KW11-L VECTOR STORAGE
003152  000104          PCVEC:  .WORD   104             ;KW11-P VECTOR STORAGE
        000114          MEMVEC=114                      ;MEMORY PARITY TRAP VECTOR
003154  000000          TCONLO: .WORD   0               ;LO BITS OF CALIBRATION TIME CONSTANT
003156  000000          TCONHI: .WORD   0               ;HI BITS OF CALIB TIME CONST
003160  000000          BLWMIN: .WORD   0               ;COUNT OF TIMES BELOW MIN
003162  000000          ABVMX1: .WORD   0               ;COUNT OF FORWARD TIMES ABOVE MAX
003164  000000          ABVMX2: .WORD   0               ;COUNT OF REVERSE TIMES ABOVE MAX
003166  000000          SUMLO1: .WORD   0               ;LO BITS OF FORWARD TIME SUM
003170  000000          SUMHI1: .WORD   0               ;HI BITS OF FORWARD TIME SUM
003172  000000          SUMLO2: .WORD   0               ;LO BITS OF REVERSE TIME SUM
003174  000000          SUMHI2: .WORD   0               ;HI BITS OF REVERSE TIME SUM
003176  000000          SAVPAR: .WORD   0               ;SAVE WORD FOR PAR CONSTANT
003200  000000          SAVWRD: .WORD   0               ;SCRATCH WORD
003202  000010          CYLLST: .BLKW   ^D8             ;LIST OF PSEUDO-RAND CYL ADDRESSES
003222  000400          BSSOFT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM SOFTWARE
004222  000400          BSFACT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM FACTORY
005222  000006          PRVCMD: .BLKW   6               ;PREVIOUS COMMAND STORAGE
005236  000006          COMSTR: .BLKW   6               ;CURRENT COMMAND STORAGE
005252  000000          PRMPLO: .WORD   0               ;PREV. U.B. MAP REG 0
005254  000000          PRMPHO: .WORD   0
005256  000000          CRMPLO: .WORD   0               ;CURRENT U.B. MAP REG 0
005260  000000          CRMPHO: .WORD   0
005262  000102          BUFFO:  .BLKW   ^D66            ;OUTPUT BUFFER 1
005466  000000          LOWOCT: .WORD   0               ;LOW 16 BITS OF CONVERTED BINARY NUMBER
005470  000000          HIGOCT: .WORD   0               ;HIGH BITS OF CONVERTED BINARY NO.
005472  000000          BUFPRT: .WORD   0               ;BUFFER POINTER
005474  000000          RECODE: .WORD   0               ;RECOVERY CODE WORD
005476  000000          ERRCOM: .WORD   0               ;ERROR COMMAND
005500  000000          DRIVE:  .WORD   0               ;NO.  OF  DRIVE IN USE
005502  000000          STALLS: .WORD   0               ;CURRENT NO. OF UNIT STALLS TO APPLY
005504  000000          CYLNDR: .WORD   0               ;CURRENT CYLINDER NUMBER
005506  000000          FS:     .WORD   0               ;FIRST SECTOR LIMIT
005510  000000          LS:     .WORD   0               ;LAST SECTOR LIMIT
005512  000000          NCYL1:  .WORD   0               ;NEXT CYL SCRATCH WORD
005514  000000          NCYL2:  .WORD   0               ;NEXT CYL SCRATCH WORD
005516  000000          OFINUS: .WORD   0               ;OFFSET IN USE
005520  000000          TRACK:  .WORD   0               ;TRACK IN USE
```

```
3729  005522  000000  INTCHR: .WORD   0               ;TTY INTERRUPT INPUT WORD
3730  005524  000000  SELECT: .WORD   0               ;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3731  005526  000000  ONLINE: .WORD   0               ;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3732  005530  000000  NEWON:  .WORD   0               ;NEW LOOK AT ONLINE DRIVES
3733  005532  000000  SCRACH: .WORD   0               ;ALL-PURPOSE SCRATCH WORD
3734  005534  000000  PATRN:  .WORD   0               ;DATA PATTERN LIST WORD
3735  005536  000000  XXDPAD: .WORD   0               ;STARTING ADDRESS OF XXDP LOADER
3736  005540  000002  XDPSAV: .BLKW   2               ;XXDP PHYSICAL SAVE ADDRESS
3737  005544  000000  PLOFST: .WORD   0               ;(+) OFFSET VALUE
3738  005546  000000  NGOFST: .WORD   0               ;(-) OFFSET VALUE
3739  005550  000005  SAVPRS: .BLKW   5               ;SAVE INITIAL PARAMS FOR XFER
3740  005562  000000  WDSXFR: .WORD   0               ;NO. OF WORDS ACTUALLY XFERRED
3741  005564  000000  FINCYL: .WORD   0               ;FINAL CYLINDER ADRS
3742  005566     000  FINTRK: .BYTE   0               ;FINAL TRACK ADRS
3743  005567     000  FINSEC: .BYTE   0               ;FINAL SECTOR ADRS
3744  005570  000000  LASTWC: .WORD   0               ;ACTUAL FINAL WC
3745  005572  157776  MAHILM: .WORD   157776          ;MA UPPER LIMIT
3746  005574  000077          .WORD   77
3747  005576  064640  MA:     .WORD   RWBUF           ;LO BITS OF PHYS MEM ADRS
3748  005600  000000          .WORD   0               ;HI BITS OF PHYS MEM ADRS
3749  005602  000002  PMA:     .BLKW   2               ;PARTIAL TRANSFER MEMORY START ADRS
3750
3751
3752
3753                          ;LIST OF DESIRED DRIVES TO TEST (IF BYTE=0, DON'T TEST DRIVE)
3754  005606                  DRVLST:
3755  005606     001                  .BYTE   1       ;DRIVE 0
3756  005607     001                  .BYTE   1       ;DRIVE 1
3757  005610     001                  .BYTE   1       ;DRIVE 2
3758  005611     001                  .BYTE   1       ;DRIVE 3
3759  005612     001                  .BYTE   1       ;DRIVE 4
3760  005613     001                  .BYTE   1       ;DRIVE 5
3761  005614     001                  .BYTE   1       ;DRIVE 6
3762  005615     001                  .BYTE   1       ;DRIVE 7
3763
3764
3765
3766
3767                          ;LIST OF ITERATION COUNTS FOR DESIRED TESTS (IF=0, TEST NOT RUN)
                             ;MAXIMUM ALLOWABLE NUMBER OF ITERATIONS = 77776 (OCT)
3768  005616                  TSTLST:
3769  005616  000002                .WORD   2       ;TEST 1
3770  005620  000002                .WORD   2       ;TEST 2
3771  005622  000002                .WORD   2       ;TEST 3
3772  005624  000002                .WORD   2       ;TEST 4
3773  005626  000002                .WORD   2       ;TEST 5
3774  005630  000100                .WORD   100     ;TEST 6
3775
3776
3777
3778
3779  005632                  ;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
                             DFLTST:
3780  005632  000002                .WORD   2       ;TEST 1
3781  005634  000002                .WORD   2       ;TEST 2
3782  005636  000002                .WORD   2       ;TEST 3
3783  005640  000002                .WORD   2       ;TEST 4
3784  005642  000002                .WORD   2       ;TEST 5
```

```
                        PROGRAM SPECIFIC RESERVED LOCATIONS

3785  005644  000100                .WORD   100             ;TEST 6
3786
3787
3788          000006    NMTSTS=<DFLTST-TSTLST>/2          ;TOTAL NO. OF AUTOMATIC TESTS
3789
3790
3791                    ;OPERATING PARAMETER LIST
3792  005646            PRMLST:
3793  005646  000000    FC:     .WORD   0               ;FIRST CYLINDER
3794  005650  000632    LC:     .WORD   632             ;LAST CYLINDER
3795  005652  000000    FT:     .WORD   0               ;FIRST TRACK
3796  005654  000002    LT:     .WORD   2               ;LAST TRACK
3797  005656  000000    S0:     .WORD   0               ;FIRST SECTOR IF 20(DEC) SECTOR FMT
3798  005660  000023    S1:     .WORD   23              ;LAST SECTOR IF 20(DEC) SECTOR FMT
3799  005662  000000    S2:     .WORD   0               ;FIRST SECTOR IF 22(DEC) SECTOR FMT
3800  005664  000025    S3:     .WORD   25              ;LAST SECTOR IF 22(DEC) SECTOR FMT
3801  005666  000000    PT:     .WORD   0               ;DATA PATTERN SELECT WORD
3802  005670  000000    CS:     .WORD   0               ;CONTROL SWITCH WORD
3803  005672  000000    ST:     .WORD   0               ;NUMBER OF UNIT STALLS
3804
3805
3806                    ;DEFAULT OPERATING PARAMETER VALUES
3807  005674            PRDFLT:
3808  005674  000000            .WORD   0               ;FC DEFAULT
3809  005676  000632            .WORD   632             ;LC DEFAULT
3810  005700  000000            .WORD   0               ;FT DEFAULT
3811  005702  000002            .WORD   2               ;LT DEFAULT
3812  005704  000000            .WORD   0               ;S0 DEFAULT
3813  005706  000023            .WORD   23              ;S1 DEFAULT
3814  005710  000000            .WORD   0               ;S2 DEFAULT
3815  005712  000025            .WORD   25              ;S3 DEFAULT
3816  005714  000000            .WORD   0               ;PT DEFAULT
3817  005716  000000            .WORD   0               ;CS DEFAULT
3818  005720  000000            .WORD   0               ;ST DEFAULT
3819
3820
3821
3822
                        ;OPERATING PARAMETER VALUE LOW AND HIGH LIMITS
3823  005722            PRMLIM:
3824  005722  000000            .WORD   0               ;FC LIMITS
3825  005724  000631            .WORD   631
3826  005726  000000            .WORD   0               ;LC LIMITS
3827  005730  000632            .WORD   632
3828  005732  000000            .WORD   0               ;FT LIMITS
3829  005734  000002            .WORD   2
3830  005736  000000            .WORD   0               ;LT LIMITS
3831  005740  000002            .WORD   2
3832  005742  000000            .WORD   0               ;S0 LIMITS
3833  005744  000023            .WORD   23
3834  005746  000000            .WORD   0               ;S1 LIMITS
3835  005750  000023            .WORD   23
3836  005752  000000            .WORD   0               ;S2 LIMITS
3837  005754  000025            .WORD   25
3838  005756  000000            .WORD   0               ;S3 LIMITS
3839  005760  000025            .WORD   25
3840  005762  000000            .WORD   0               ;PT LIMITS
```

```
3841  005764  177777                .WORD   177777
3842  005766  000000                .WORD   0               ;CS LIMITS
3843  005770  000062                .WORD   000062
3844  005772  000000                .WORD   0               ;ST LIMITS
3845  005774  177777                .WORD   177777
3846
3847
3848
3849                                ;ASCII PARAMETER MNEMONICS
3850  005776                        PRMNEM:
3851  005776  041506                .ASCII  /FC/
3852  006000  041514                .ASCII  /LC/
3853  006002  052106                .ASCII  /FT/
3854  006004  052114                .ASCII  /LT/
3855  006006  030123                .ASCII  /S0/
3856  006010  030523                .ASCII  /S1/
3857  006012  031123                .ASCII  /S2/
3858  006014  031523                 ASCII  /S3/
3859  006016  052120                .ASCII  /PT/
3860  006020  051503                .ASCII  /CS/
3861  006022  052123                .ASCII  /ST/
3862
3863
3864
3865                                ;DATA PATTERN 00
3866                                ;       HI-LO FREQ. MIX
3867  006024                        PAT00:
3868  006024  177777                        177777
3869  006026  177777                        177777
3870  006030  177777                        177777
3871  006032  052525                        052525
3872  006034  052525                        052525
3873  006036  052525                        052525
3874  006040  177777                        177777
3875  006042  177777                        177777
3876  006044  052525                        052525
3877  006046  052525                        052525
3878  006050  177777                        177777
3879  006052  052525                        052525
3880  006054  177252                        177252
3881  006056  177252                        177252
3882  006060  172765                        172765
3883  006062  172765                        172765
3884
3885
3886
3887                                ;DATA PATTERN 01
3888                                ;       HI FREQ. PHASE MIX
3889  006064                        PAT01:
3890  006064  000000                        000000
3891  006066  000000                        000000
3892  006070  000000                        000000
3893  006072  177777                        177777
3894  006074  177777                        177777
3895  006076  177777                        177777
3896  006100  000000                        000000
```

```
  3897  006102  000000                  000000
  3898  006104  177777                  177777
  3899  006106  177777                  177777
  3900  006110  000000                  000000
  3901  006112  177777                  177777
  3902  006114  000000                  000000
  3903  006116  177777                  177777
  3904  006120  000000                  000000
  3905  006122  177777                  177777
  3906
  3907
  3908
  3909                                ;DATA PATTERN 02
  3910                                ;       LO FREQ. PHASE MIX
  3911  006124                        PAT02:
  3912  006124  052525                  052525
  3913  006126  052525                  052525
  3914  006130  052525                  052525
  3915  006132  125252                  125252
  3916  006134  125252                  125252
  3917  006136  125252                  125252
  3918  006140  052525                  052525
  3919  006142  052525                  052525
  3920  006144  125252                  125252
  3921  006146  125252                  125252
  3922  006150  052525                  052525
  3923  006152  125252                  125252
  3924  006154  052525                  052525
  3925  006156  125252                  125252
  3926  006160  052525                  052525
  3927  006162  125252                  125252
  3928
  3929
  3930
  3931                                ;DATA PATTERN 03
  3932                                ;       MAX. PRECOMP. PHASE MIX
  3933  006164                        PAT03:
  3934  006164  133333                  133333
  3935  006166  066666                  066666
  3936  006170  155555                  155555
  3937  006172  155555                  155555
  3938  006174  133333                  133333
  3939  006176  066666                  066666
  3940  006200  066666                  066666
  3941  006202  155555                  155555
  3942  006204  155555                  155555
  3943  006206  133333                  133333
  3944  006210  133333                  133333
  3945  006212  133333                  133333
  3946  006214  133333                  133333
  3947  006216  133333                  133333
  3948  006220  133333                  133333
  3949  006222  133333                  133333
  3950
  3951
  3952
```

```
                                ;DATA PATTERN 04
                                ;       ROTATING BOUNDARY PULSE PRECOMP.
                                PAT04:
006224                                  121105
006224  121105                          150442
006226  150442                          064221
006230  064221                          132110
006232  132110                          055044
006234  055044                          026422
006236  026422                          013211
006240  013211                          105504
006242  105504                          042642
006244  042642                          021321
006246  021321                          110550
006250  110550                          044264
006252  044264                          022132
006254  022132                          011055
006256  011055                          104426
006260  104426                          042213
006262  042213

                                ;DATA PATTERN 05
                                ;       ROTATING CELL PULSE PRECOMP.
                                PAT05:
006264                                  026455
006264  026455                          113226
006266  113226                          045513
006270  045513                          122645
006272  122645                          151322
006274  151322                          064551
006276  064551                          132264
006300  132264                          055132
006302  055132                          026455
006304  026455                          113226
006306  113226                          045513
006310  045513                          122645
006312  122645                          151322
006314  151322                          064551
006316  064551                          132264
006320  132264                          055132
006322  055132

                                ;DATA PATTERN 06
                                ;       ALL ZEROS
                                PAT06:
006324                                  000000
006324  000000                          000000
006326  000000                          000000
006330  000000                          000000
006332  000000                          000000
006334  000000                          000000
006336  000000                          000000
006340  000000                          000000
006342  000000                          000000
006344  000000                          000000
006346  000000                          000000
```

```
4009  006350  000000                    000000
4010  006352  000000                    000000
4011  006354  000000                    000000
4012  006356  000000                    000000
4013  006360  000000                    000000
4014  006362  000000                    000000
4015
4016
4017
4018                                    ;DATA PATTERN 07
4019                                    ;       ALL ONES
4020  006364                            PAT07:
4021  006364  177777                    177777
4022  006366  177777                    177777
4023  006370  177777                    177777
4024  006372  177777                    177777
4025  006374  177777                    177777
4026  006376  177777                    177777
4027  006400  177777                    177777
4028  006402  177777                    177777
4029  006404  177777                    177777
4030  006406  177777                    177777
4031  006410  177777                    177777
4032  006412  177777                    177777
4033  006414  177777                    177777
4034  006416  177777                    177777
4035  006420  177777                    177777
4036  006422  177777                    177777
4037
4038
4039                                    ;DATA PATTERN 08
4040                                    ;       SHIFTED 1 IN FIELD OF ZEROS
4041  006424                            PAT08:
4042  006424  000001                    000001
4043  006426  000002                    000002
4044  006430  000004                    000004
4045  006432  000010                    000010
4046  006434  000020                    000020
4047  006436  000040                    000040
4048  006440  000100                    000100
4049  006442  000200                    000200
4050  006444  000400                    000400
4051  006446  001000                    001000
4052  006450  002000                    002000
4053  006452  004000                    004000
4054  006454  010000                    010000
4055  006456  020000                    020000
4056  006460  040000                    040000
4057  006462  100000                    100000
4058
4059
4060
4061                                    ;DATA PATTERN 09
4062                                    ;       SHIFTED 0 IN FIELD OF ONES
4063  006464                            PAT09:
4064  006464  177776                    177776
```

D07


```
006466  177775                          177775
006470  177773                          177773
006472  177767                          177767
006474  177757                          177757
006476  177737                          177737
006500  177677                          177677
006502  177577                          177577
006504  177377                          177377
006506  176777                          176777
006510  175777                          175777
006512  173777                          173777
006514  167777                          167777
006516  157777                          157777
006520  137777                          137777
006522  077777                          077777



                                ;DATA PATTERN 10
                                ;       ALTERNATING 0-1
006524                          PAT10:
006524  052525                          052525
006526  052525                          052525
006530  052525                          052525
006532  052525                          052525
006534  052525                          052525
006536  052525                          052525
006540  052525                          052525
006542  052525                          052525
006544  052525                          052525
006546  052525                          052525
006550  052525                          052525
006552  052525                          052525
006554  052525                          052525
006556  052525                          052525
006560  052525                          052525
006562  052525                          052525



                                ;DATA PATTERN 11
                                ;       ALTERNATING 1-0
006564                          PAT11:
006564  125252                          125252
006566  125252                          125252
006570  125252                          125252
006572  125252                          125252
006574  125252                          125252
006576  125252                          125252
006600  125252                          125252
006602  125252                          125252
006604  125252                          125252
006606  125252                          125252
006610  125252                          125252
006612  125252                          125252
006614  125252                          125252
```

```
004121  006616  125252                  125252
004122  006620  125252                  125252
004123  006622  125252                  125252
004124
004125
004126
004127
004128                          ;DATA PATTERN 12
004129                          ;       SHIFTING ZEROS AND ONES
004130  006624                  PAT12:
004131  006624  000001                  000001
004132  006626  000003                  000003
004133  006630  000007                  000007
004134  006632  000017                  000017
004135  006634  000037                  000037
004136  006636  000077                  000077
004137  006640  000177                  000177
004138  006642  000377                  000377
004139  006644  000777                  000777
004140  006646  001777                  001777
004141  006650  003777                  003777
004142  006652  007777                  007777
004143  006654  017777                  017777
004144  006656  037777                  037777
004145  006660  077777                  077777
004146  006662  177777                  177777
004147
004148
004149                          ;DATA PATTERN 13
004150                          ;       COMPOSITE ROTATING
004151  006664                  PAT13:
004152  006664  072307                  072307
004153  006666  135143                  135143
004154  006670  156461                  156461
004155  006672  167230                  167230
004156  006674  073514                  073514
004157  006676  035646                  035646
004158  006700  016723                  016723
004159  006702  107351                  107351
004160  006704  143564                  143564
004161  006706  061672                  061672
004162  006710  030735                  030735
004163  006712  114356                  114356
004164  006714  046167                  046167
004165  006716  123073                  123073
004166  006720  151453                  151453
004167  006722  164616                  164616
004168
004169
004170
004171                          ;DATA PATTERN 14
004172                          ;       PSEUDO-RANDOM (COMPUTED BY PROGRAM)
004173  006724                  PAT14:
004174  006724  000000                  .WORD
004175  006726  000000                  .WORD
004176  006730  000000                  .WORD
```

```
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD
                                        .WORD


                                ;USER-DEFINED DATA PATTERN 15
                                PAT15:
                                        072307                  ;WORD 00
                                        135143                  ;WORD 01
                                        156461                  ;WORD 02
                                        167230                  ;WORD 03
                                        073514                  ;WORD 04
                                        035645                  ;WORD 05
                                        016723                  ;WORD 06
                                        107351                  ;WORD 07
                                        143564                  ;WORD 10
                                        061672                  ;WORD 11
                                        030735                  ;WORD 12
                                        114356                  ;WORD 13
                                        046167                  ;WORD 14
                                        123073                  ;WORD 15
                                        151453                  ;WORD 16
                                        164616                  ;WORD 17



                        057467  MINLO1=^D24375          ;LO BITS OF SPEC'D MIN ROT. LATENCY
                        000000  MINHI1=0                ;HI BITS OF SPEC'D MIN ROT. LATENCY
                        062031  MAXLO1=^D25625          ;LO BITS OF SPEC'D MAX ROT. LATENCY
                        000000  MAXHI1=0                ;HI BITS OF SPEC'D MAX ROT. LATENCY
                        017500  MAXLO2=^D8000           ;LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
                        000000  MAXHI2=0                ;HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
                        112160  MAXLO3=^D38000          ;LO BITS OF SPEC'D MAX AVG SEEK TIME
                        000000  MAXHI3=0                ;HI BITS OF SPEC'D MAX AVG SEEK TIME
                        022370  MAXLO4=^D9464           ;LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
                        000001  MAXHI4=1                ;HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
                        002734  RNDSHT=^D1500           ;SHORT RANDOM SEEKS = 1500(10)
                        016514  RNDLNG=^D7500           ;LONG RANDOM SEEKS = 7500(10)
                        000632  LSTCYL=632              ;LAST CYL. = 410(10)
                        000002  LSTTRK=2                ;LAST TRACK  = 2
                        000365  ALNCYL=365              ;ALIGNMENT CYLINDER =245(10)
                        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

        007024   005015  042115  030455     DZR6N:  .ASCIZ  <15><12>/MD-11-DZR6N-C/
        007032   026461  055104  033122
        007040   026516  000103
        007044   026440  051040  033113     SUBVER: .ASCIZ  & - RK611/RK06 SUBSYSTEM VERIFICATION : PART &
        007052   030461  051057  030113
        007060   020066  052523  051502
        007066   051531  042524  020115
        007074   042526  044522  044506
        007102   040503  044524  047117
        007110   035040  050040  051101
        007116   020124     000
        007121      061  005015     000     PART1:  .ASCIZ  /1/<15><12>
        007125      062  005015     000     PART2:  .ASCIZ  /2/<15><12>
        007131      015  005012  040514     LSTMEM: .ASCIZ  <15><12><12>/LAST PHYS MEM ADR = /
        007136   052123  050040  054510
        007144   020123  042515  020115
        007152   042101  020122  020075
        007160      000
        007161      117  042526  046122     OVLODR: .ASCIZ  /OVERLAY LOADER ? (Y OR N) * /
        007166   054501  046040  040517
        007174   042504  020122  020077
        007202   054450  047440  020122
        007210   024516  025040  000040
        007216   005015  040520  040522     PRMINP: .ASCIZ  <15><12>/PARAMETER INPUT MODE/<15><12><12>
        007224   042515  042524  020122
        007232   047111  052520  020124
        007240   047515  042504  005015
        007246   000012
        007250   045522  033060  041040     RKBADR: .ASCIZ  /RK06 BUS ADR = /
        007256   051525  040440  051104
        007264   036440  000040
        007270   045522  033060  053040     RKVADR: .ASCIZ  /RK06 VEC ADR = /
        007276   041505  040440  051104
        007304   036440  000040
        007310   045522  033060  050040     RKPRTY: .ASCIZ  /RK06 PRIORITY =/
        007316   044522  051117  052111
        007324   020131  000075
        007330   053523  020122  020075     SWRMSG: .ASCIZ  /SWR = /
        007336      000
        007337      040  047040  053505     NEWMSG: .ASCIZ  /  NEW = /
        007344   036440  000040
        007350   005015  051104  053111     DRVSEQ: .ASCIZ  <15><12>/DRIVE(S) = /
        007356   024105  024523  036440
        007364   000040
        007366   051104  053111  020105     BADDRV: .ASCIZ  /DRIVE   /
        007374   020040     000
        007377      116  047117  042455     NXDRIV: .ASCIZ  /NON-EXISTENT/<15><12>
```

```
 4289  007404  044530  052123  047105
 4290  007412  006524  000012
 4291  007416  047516  020124  042522  NTREDY: .ASCIZ  /NOT READY/<15><12>
 4292  007424  042101  006531  000012
 4293  007432  051127  052111  026505  WRTLOK: .ASCIZ  /WRITE-LOCKED/<15><12>
 4294  007440  047514  043503  042105
 4295  007446  005015     000
 4296  007451     114  040517  042504  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
 4297  007456  020104  044527  044124
 4298  007464  040440  044514  047107
 4299  007472  050040  041501  006513
 4300  007500  000012
 4301  007502  043111  054040  042130  REPLPK: .ASCIZ  /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", &  REPLACE IT : /
 4302  007510  020120  040520  045503
 4303  007516  047440  020116  051104
 4304  007524  020126  026060  052040
 4305  007532  050131  020105  054442
 4306  007540  036040  051103  021076
 4307  007546  020054  020046  051040
 4308  007554  050105  040514  042503
 4309  007562  044440  020124  020072
 4310  007570     000
 4311  007571     015  005012  025052  NODRTS: .ASCII  <15><12><12>/** NO DRIVES TO TEST/<15><12>
 4312  007576  047040  020117  051104
 4313  007604  053111  051505  052040
 4314  007612  020117  042524  052123
 4315  007620  005015
 4316  007622  051120  051505  020123  CNTRDY: .ASCIZ  /PRESS "CONT" WHEN RDY/<15><12>
 4317  007630  041442  047117  021124
 4318  007636  053440  042510  020116
 4319  007644  042122  006531  000012
 4320  007652  040510  052114  051040  HLTRQD: .ASCIZ  /HALT REQUESTED/<15><12>
 4321  007660  050505  042525  052123
 4322  007666  042105  005015     000
 4323  007673     104  044522  042526  DROXDP: .ASCIZ  /DRIVE 0 IS LOAD MEDIUM/<15><12>
 4324  007700  030040  044440  020123
 4325  007706  047514  042101  046440
 4326  007714  042105  052511  006515
 4327  007722  000012
 4328  007724  005015  052012  020117  ALDRVS: .ASCIZ  <15><12><12>/TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR>/<15><12>/* /
 4329  007732  042524  052123  040440
 4330  007740  046114  042040  044522
 4331  007746  042526  020123  054524
 4332  007754  042520  021040  021101
 4333  007762  036040  051103  026076
 4334  007770  042440  051514  020105
 4335  007776  041474  037122  005015
 4336  010004  020052     000
 4337  010007     015  046012  036440  TSTMDS: .ASCII  <15><12>/L = LIST TESTS/<15><12>
 4338  010014  046040  051511  020124
 4339  010022  042524  052123  006523
 4340  010030     012
 4341  010031     103  036440  041440          .ASCII  /C = CHANGE TESTS/<15><12>
 4342  010036  040510  043516  020105
 4343  010044  042524  052123  006523
 4344  010052     012
```

```
 4345  010053     111  036440  044440          .ASCIZ  /I = INPUT PARAMS, RUN TESTS/<15><12>
 4346  010060  050116  052125  050040
 4347  010066  051101  046501  026123
 4348  010074  051040  047125  052040
 4349  010102  051505  051524  005015
 4350  010110     000
 4351  010111     015  042412  052116  ENTLCI: .ASCIZ  <15><12>/ENTER L,C, OR I/<15><12>/* /
 4352  010116  051105  046040  041454
 4353  010124  020054  051117  044440
 4354  010132  005015  020052     000
 4355  010137     124  020117  042504  DFTEST: .ASCIZ  /TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12>/* /
 4356  010144  040506  046125  020124
 4357  010152  042524  052123  020123
 4358  010160  054524  042520  042040
 4359  010166  036040  051103  026076
 4360  010174  042440  051514  020105
 4361  010202  041474  037122  005015
 4362  010210  020052     000
 4363  010213     015  052012  051505  TLSTHD: .ASCIZ  <15><12>/TEST     ITERATIONS/<15><12>
 4364  010220  020124  020040  020040
 4365  010226  052111  051105  052101
 4366  010234  047511  051516  005015
 4367  010242     000
 4368  010243     015  052012  036440  EXPLAN: .ASCII  <15><12>/T = TYPE PARAM LIST/<15><12>
 4369  010250  052040  050131  020105
 4370  010256  040520  040522  020115
 4371  010264  044514  052123  005015
 4372  010272  020117  020075  050117          .ASCII  /O = OPEN PARAM LIST/<15><12>
 4373  010300  047105  050040  051101
 4374  010306  046501  046040  051511
 4375  010314  006524     012
 4376  010317     123  036440  051440          .ASCII  /S = SET INDIVIDUAL PARAM/<15><12>
 4377  010324  052105  044440  042116
 4378  010332  053111  042111  040525
 4379  010340  020114  040520  040522
 4380  010346  006515     012
 4381  010351     122  036440  051040          .ASCIZ  /R = RUN TESTS/<15><12>
 4382  010356  047125  052040  051505
 4383  010364  051524  005015     000
 4384  010371     015  042412  052116  PARMDE: .ASCIZ  <15><12>/ENTER T,O,S, OR R/<15><12>
 4385  010376  051105  052040  047454
 4386  010404  051454  020054  051117
 4387  010412  051040  005015     000
 4388  010417     015  025012  042040  DUACES: .ASCIZ  <15><12>/* DUAL-ACCESS DATA TEST */<15><12>
 4389  010424  040525  026514  041501
 4390  010432  042503  051523  042040
 4391  010440  052101  020101  042524
 4392  010446  052123  025040  005015
 4393  010454     000
 4394  010455     115  054101  053440  MAWRDC: .ASCIZ  /MAX WORD COUNT = /
 4395  010462  051117  020104  047503
 4396  010470  047125  020124  020075
 4397  010476     000
 4398  010477     052  020052  051440  SECNL1: .ASCII  /**  SO>S1/
 4399  010504  037060  030523
 4400  010510  047040  052117  040440  NOTALD: .ASCIZ  / NOT ALLOWED/<15><12>
```

```
 4401  010516  046114  053517  042105
 4402  010524  005015     000
 4403  010527     052  020052  051440  SECNL2: .ASCIZ  /** S2>S3/
 4404  010534  037063  031523     000
 4405  010541     052  020052  043040  TRKNLW: .ASCIZ  /** FT>LT/
 4406  010546  037124  052114     000
 4407  010553     052  020052  053440  WC2BIG: ,ASCIZ  /** WC OR MA TOO LARGE/<15><12>
 4408  010560  020103  051117  046440
 4409  010566  020101  047524  020117
 4410  010574  040514  043522  006505
 4411  010602  000012
 4412  010604  047524  042040  043105  DFQUES: .ASCIZ  /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR>/<15><12>/* /
 4413  010612  052501  052114  040440
 4414  010620  046114  050040  051101
 4415  010626  046501  026123  052040
 4416  010634  050131  020105  020104
 4417  010642  041474  037122  020054
 4418  010650  046105  042523  036040
 4419  010656  051103  006476  025012
 4420  010664  000040
 4421  010666  005015  051525  051105  PFIFTN: .ASCIZ  <15><12>/USER-DEFINED PATTERN 15 :/<15><12>
 4422  010674  042055  043105  047111
 4423  010702  042105  050040  052101
 4424  010710  042524  047122  030440
 4425  010716  020065  006472  000012
 4426  010724  005015  047515  044504  SELP15: .ASCIZ  <15><12>/MODIFY PATTERN 15 :/<15><12>
 4427  010732  054506  050040  052101
 4428  010740  042524  047122  030440
 4429  010746  020065  006472  000012
 4430  010754  047524  046440  042117  MDFY15: .ASCIZ  /TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
 4431  010762  043111  020131  040520
 4432  010770  052124  051105  020116
 4433  010776  032461  020054  054524
 4434  011004  042520  046440  036040
 4435  011012  051103  026076  042440
 4436  011020  051514  020105  041474
 4437  011026  037122  005015  020052
 4438  011034     000
 4439  011035     015  042412  052116  ENTPAS: .ASCIZ  <15><12>/ENTER NO. OF PASSES (1-77777) :/<15><12>/* /
 4440  011042  051105  047040  027117
 4441  011050  047440  020106  040520
 4442  011056  051523  051505  024040
 4443  011064  026461  033467  033467
 4444  011072  024467  035040  005015
 4445  011100  020052     000
 4446  011103     015  025012  020052  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRIVE - 20 ERRORS/<15><12>
 4447  011110  042040  047522  050120
 4448  011116  047111  020107  051104
 4449  011124  053111  020105  020055
 4450  011132  030062  042440  051122
 4451  011140  051117  006523  000012
 4452  011146  005015  052012  051505  TSTDRN: .ASCII  <15><12><12>/TESTING DRIVE /
 4453  011154  044524  043516  042040
 4454  011162  044522  042526     040
 4455  011167     040  005015     000  DRVNO:  .ASCIZ  / /<15><12>
 4456  011173     104  044522  042526  DRIV:   .ASCIZ  /DRIVE /
```

```
4457  011200  000040
4458  011202  040503  052122  020056  CART:   .ASCIZ  /CART. /
4459  011210     000
4460  011211     123  051105  020056  SERNM:  .ASCIZ  /SER. NO.  /
4461  011216  047516  020056  000040
4462  011224  005015  040506  052103  FACTBS: .ASCIZ  <15><12>/FACTORY /
4463  011232  051117  020131     000
4464  011237     012  047523  052106  SOFTBS: .ASCII  <12>/SOFTWARE /
4465  011244  040527  042522     040
4466  011251     102  042101  051440  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
4467  011256  041505  047524  051522
4468  011264  035040  005015     000
4469  011271     040  047040  047117  NOFALS: .ASCIZ  /  NONE/
4470  011276  000105
4471
4472
4473                                  ;MESSAGES USED IN OFFSET-TO-FAILURE TEST
4474  011300  005015  047412  043106  OFSHED: .ASCII  <15><12><12>/OFFSET-TO-FAILURE MEASUREMENTS :/<15><12><12>
4475  011306  042523  026524  047524
4476  011314  043055  044501  052514
4477  011322  042522  046440  040505
4478  011330  052523  042522  042515
4479  011336  052116  020123  006472
4480  011344  005012
4481  011346  051124  041501  020113          .ASCII  /TRACK  CYLN  SECT  +OFST  -OFST/<15><12>
4482  011354  041440  046131  020116
4483  011362  051440  041505  020124
4484  011370  025440  043117  052123
4485  011376  020040  047455  051506
4486  011404  006524     012
4487  011407     011  020011  020040          .ASCIZ  <011><011>/   (UIN)  (UIN)/<15><12>
4488  011414  052450  047111  020051
4489  011422  024040  044525  024516
4490  011430  005015     000
4491
4492
4493  011433     000                  UNLOD:  .BYTE   0               ;DRIVE UNLOADED INDICATOR
4494  011434     000                  VERIFY: .BYTE   0               ;VERIFY MODE FLAG
4495  011435     015  005012  025052  IDENT:  .ASCIZ  <15><12><12>/*** RK06 HEAD ALIGNMENT AID ***/<15><12>
4496  011442  020052  045522  033060
4497  011450  044040  040505  020104
4498  011456  046101  043511  046516
4499  011464  047105  020124  044501
4500  011472  020104  025052  006452
4501  011500  000012
4502  011502  047506  020122  042510  FORHLP: .ASCIZ  /FOR HELP TYPE H, ELSE <CR>/<15><12>
4503  011510  050114  052040  050131
4504  011516  020105  026110  042440
4505  011524  051514  020105  041474
4506  011532  037122  005015     000
4507  011537     012  040515  052516  TYPMOD: .ASCIZ  <12>/MANUAL OR AUTO MODE (M OR A)?/<15><12>
4508  011544  046101  047440  020122
4509  011552  052501  047524  046440
4510  011560  042117  020105  046450
4511  011566  047440  020122  024501
4512  011574  006477  000012
```

```
 4513  011600  005012  020052  040515  TYPMAN: .ASCIZ  <12><12>/* MANUAL SELECT MODE */<15><12>
 4514  011606  052516  046101  051440
 4515  011614  046105  041505  020124
 4516  011622  047515  042504  025040
 4517  011630  005015     000
 4518  011633     012  047105  042524  ENTDRV: .ASCIZ  <12>/ENTER DRIVE NO. (0-7):/<15><12>
 4519  011640  020122  051104  053111
 4520  011646  020105  047516  020056
 4521  011654  030050  033455  035051
 4522  011662  005015     000
 4523  011665     052  020052  042040  NOTRDY: .ASCII  /**  DRIVE /
 4524  011672  044522  042526     040
 4525  011677     040  047040  052117  DRNRDY: .ASCII  /  NOT READY !  START DRIVE IF NECESSARY./<15><12>
 4526  011704  051040  040505  054504
 4527  011712  020440  020040  052123
 4528  011720  051101  020124  051104
 4529  011726  053111  020105  043111
 4530  011734  047040  041505  051505
 4531  011742  040523  054522  006456
 4532  011750     012
 4533  011751     040  020040  053440          .ASCIZ  /    WHEN DRIVE IS READY, PRESS "CONT" ON CPU./<15><12>
 4534  011756  042510  020116  051104
 4535  011764  053111  020105  051511
 4536  011772  051040  040505  054504
 4537  012000  020054  051120  051505
 4538  012006  020123  041442  047117
 4539  012014  021124  047440  020116
 4540  012022  050103  027125  005015
 4541  012030     000
 4542  012031     052  020052  042040  NONEXD: .ASCII  /**  DRIVE /
 4543  012036  044522  042526     040
 4544  012043     040  047040  047117  DRVNED: .ASCIZ  /  NON-EXISTENT OR OFF-LINE ! PRESS "CONT" TO RESTART./<15><12>
 4545  012050  042455  044530  052123
 4546  012056  047105  020124  051117
 4547  012064  047440  043106  046055
 4548  012072  047111  020105  020041
 4549  012100  051120  051505  020123
 4550  012106  041442  047117  021124
 4551  012114  052040  020117  042522
 4552  012122  052123  051101  027124
 4553  012130  005015     000
 4554  012133     040  020040  041440  HALTER: .ASCIZ  /    CPU WILL HALT./<15><12>
 4555  012140  052520  053440  046111
 4556  012146  020114  040510  052114
 4557  012154  006456  000012
 4558  012160  025052  020040  051104  NOTLOK: .ASCII  /**  DRIVE /
 4559  012166  053111  020105
 4560  012172  020040  047516  020124  NODRLK: .ASCII  /  NOT WRITE-LOCKED ! PLEASE SET WRITE LOCK SWITCH ./<15><12>
 4561  012200  051127  052111  026505
 4562  012206  047514  045503  042105
 4563  012214  020440  050040  042514
 4564  012222  051501  020105  042523
 4565  012230  020124  051127  052111
 4566  012236  020105  047514  045503
 4567  012244  051440  044527  041524
 4568  012252  020110  006456     012
```

```
4569  012257     040  020040  052040          .ASCIZ  /    THEN PRESS "CONT" ON CPU./<15><12>
4570  012264  042510  020116  051120
4571  012272  051505  020123  041442
4572  012300  047117  021124  047440
4573  012306  020116  050103  027125
4574  012314  005015     000
4575  012317     052  020052  046440  MULSEL: .ASCIZ  **  MULTIPLE DRIVES AUTO-SELECTED ! PRESS 'CONT' TO RESTART./<15><12>
4576  012324  046125  044524  046120
4577  012332  020105  051104  053111
4578  012340  051505  040440  052125
4579  012346  026517  042523  042514
4580  012354  052103  042105  020440
4581  012362  050040  042522  051523
4582  012370  023440  047503  052115
4583  012376  020047  047524  051040
4584  012404  051505  040524  052122
4585  012412  006456  000012
4586  012416  040412  044514  047107  ENTMOD: .ASCIZ  <12>/ALIGN,VERIFY, OR EXERCISE (A,V, OR E) ?/<15><12>
4587  012424  053054  051105  043111
4588  012432  026131  047440  020122
4589  012440  054105  051105  044503
4590  012446  042523  024040  026101
4591  012454  026126  047440  020122
4592  012462  024505  037440  005015
4593  012470     000
4594  012471     012  025012  051040  MANEXR: .ASCII  <12><12>/* RANDOM SEEK EXERCISES IN PROGRESS /
4595  012476  047101  047504  020115
4596  012504  042523  045505  042440
4597  012512  042530  041522  051511
4598  012520  051505  044440  020116
4599  012526  051120  043517  042522
4600  012534  051523     040
4601  012537     117  020116  051104          .ASCII  /ON DRIVE /
4602  012544  053111  020105
4603  012550  020040  006452  000012  DRIEXR: .ASCIZ  /  */<15><12>
4604  012556  005012  020052  040515  MANALN: .ASCIZ  <12><12>/* MANUAL SELECT ALIGNMENT */<15><12>
4605  012564  052516  046101  051440
4606  012572  046105  041505  020124
4607  012600  046101  043511  046516
4608  012606  047105  020124  006452
4609  012614  000012
4610  012616  005012  020052  040515  MANVRF: .ASCIZ  <12><12>/* MANUAL SELECT VERIFY */<15><12>
4611  012624  052516  046101  051440
4612  012632  046105  041505  020124
4613  012640  042526  044522  054506
4614  012646  025040  005015     000
4615  012653     110  040505  051504  HEDPOS: .ASCIZ  /HEADS POSITIONED AT CYLINDER 365 (OCT)/<15><12>
4616  012660  050040  051517  052111
4617  012666  047511  042516  020104
4618  012674  052101  041440  046131
4619  012702  047111  042504  020122
4620  012710  033063  020065  047450
4621  012716  052103  006451  000012
4622  012724  042412  052116  051105  ENTHED: .ASCIZ  <12>/ENTER HEAD NO. (0-2):/<15><12>
4623  012732  044040  040505  020104
4624  012740  047516  020056  030050
```

```
4625  012746  031055  035051  005015
4626  012754     000
4627  012755     124  050131  020105  RWNRDY: .ASCIZ  /TYPE <R> WHEN READY :/<15><12>
4628  012762  051074  020076  044127
4629  012770  047105  051040  040505
4630  012776  054504  035040  005015
4631  013004     000
4632  013005     110  040505  020104  HEDSEL: .ASCII  /HEAD /
4633  013012  020040  042523  042514  HEADNO: .ASCIZ  /  SELECTED/<15><12>
4634  013020  052103  042105  005015
4635  013026     000
4636  013027     012  025012  040440  TYPAUT: .ASCIZ  <12><12>/* AUTO SELECT MODE */<15><12>
4637  013034  052125  020117  042523
4638  013042  042514  052103  046440
4639  013050  042117  020105  006452
4640  013056  000012
4641  013060  005012  020052  052501  AUTALN: .ASCIZ  <12><12>/* AUTO SELECT ALIGNMENT */<15><12>
4642  013066  047524  051440  046105
4643  013074  041505  020124  046101
4644  013102  043511  046516  047105
4645  013110  020124  006452  000012
4646  013116  005012  020052  052501  AUTVRF: .ASCIZ  <12><12>/* AUTO SELECT VERIFY */<15><12>
4647  013124  047524  051440  046105
4648  013132  041505  020124  042526
4649  013140  044522  054506  025040
4650  013146  005015     000
4651  013151     012  051104  053111  DRISEL: .ASCII  <12>/DRIVE /
4652  013156  020105
4653  013160  020040  042523  042514  DRVSEL: .ASCIZ  /  SELECTED/<15><12><12>
4654  013166  052103  042105  005015
4655  013174  000012
4656  013176  005012  020052  052501  AUTEXR: .ASCII  <12><12>/* AUTO SELECT EXERCISES */<15><12>
4657  013204  047524  051440  046105
4658  013212  041505  020124  054105
4659  013220  051105  044503  042523
4660  013226  020123  006452     012
4661  013233     012  044123  051117          .ASCIZ  <12>/SHORT OR LONG (S OR L) ?/<15><12>
4662  013240  020124  051117  046040
4663  013246  047117  020107  051450
4664  013254  047440  020122  024514
4665  013262  037440  005015     000
4666  013267     105  042530  041522  AUTOEX: .ASCII  /EXERCISING DRIVE /
4667  013274  051511  047111  020107
4668  013302  051104  053111  020105
4669  013310  006440  000012          AUTODR: .ASCIZ  / /<15><12>
4670
4671  013314  025052  020040  040503  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRACK!/<15><12>
4672  013322  047116  052117  051040
4673  013330  040505  020104  040502
4674  013336  020104  042523  052103
4675  013344  051117  052040  040522
4676  013352  045503  006441  000012
4677  013360  041536  005015     000  CNTRLC: .ASCIZ  /↑C/<15><12>
4678  013365     136  006532  000012  CNTRLZ: .ASCIZ  /↑Z/<15><12>
4679  013372  051136  005015     000  CNTRLR: .ASCIZ  /↑R/<15><12>
4680  013377     136  006525  000012  CNTRLU: .ASCIZ  /↑U/<15><12>
```

B08


```
013404  042536  005015     000  CNTRLG: .ASCIZ  /^G/<15><12>
013411     015  005012     000  CR2LF:  .ASCIZ  <15><12><12>
013415     134     000          BKSLSH: .ASCIZ  /\/
013417     054     000          COMMA:  .ASCIZ  /,/
013421     040     040     040  SPACE6: .ASCII  /  /
013423     040                  SPACE4: .ASCII  /  /
013424     040                  SPACE3: .ASCII  / /
013425     040                  SPACE2: .ASCII  / /
013426  000040                  SPACE1: .ASCIZ  / /
                                        .EVEN
013430  020040  000075          PRMBUF: .ASCIZ  /  =/
013434  047527  042122  000040  WORDSP: .ASCIZ  /WORD /
013442  036440  000040          EQUALS: .ASCIZ  / = /
013446  020052     000          PROMPT: .ASCIZ  /* /
013451     076  000040          PRMPSP: .ASCIZ  /> /

                                        .EVEN




013454  105037  003106          DFSTRT: CLRB    MDFLAG          ;SET FLAG FOR DEFAULT MODE
013460  105037  003126                  CLRB    DULACS          ;CLEAR DUAL-ACCESS FLAG
013464  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
013470  022737  014532  000042          CMP     #DRVTST,@#42    ;SEE IF EOP RETURN ADRS = DRVTST
013476  001003                          BNE     4$              ;BR IF NOT DRVTST
013500  012737  017456  000042          MOV     #NEWPAS,@#42    ;SET RETURN ADRS = NEWPAS
013506  000405                  4$:     BR      CMSTRT          ;PROCEED


013510  112737  000001  003106  PSTART: MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
013516  105037  003126                  CLRB    DULACS          ;CLEAR DUAL-ACCESS TEST FLAG

013522  012737  000340  177776  CMSTRT: MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
013530  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
013534  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
013536  022706  001140                  CMP     #SWR,R6 ;;DONE?
013542  001374                          BNE     .-6             ;;LOOP BACK IF NO
013544  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
013550  012737  055504  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
013556  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
013564  012737  055000  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
013572  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
013600  012737  056612  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
013606  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
013614  012737  055350  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
013622  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
013630  013737  023546  023540          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
013636  005037  001304                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
013642  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
013646  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
013654  012737  013654  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
013662  012737  013662  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
```

```
                                        ;INITIALIZE THE COMMON TAGS

                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
013670 013746 000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
013674 012737 013730 000004             MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
013702 012737 177570 001140             MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
013710 012737 177570 001142             MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
013716 022777 177777 165214             CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
013724 001012                           BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
013726 000403                           BR      65$             ;;BRANCH IF NO TIMEOUT
013730 012716 013736            64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
013734 000002                           RTI
013736 012737 000176 001140     65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
013744 012737 000174 001142             MOV     #DISPREG,DISPLAY
013752 012637 000004            66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

013756 005037 001326                    CLR     $PASS           ;;CLEAR PASS COUNT
013762 132737 000200 001341             BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
013770 001403                           BEQ     67$             ;;YES,USE NON-APT SWITCH
013772 012737 001342 001140             MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
014000                          67$:
014000 000005                           RESET                   ;CLEAR THE UNIBUS
014002 012737 000006 000004             MOV     #6,@#ERRVEC     ;SET TIME-OUT VECTOR
014010 005037 000006                    CLR     @#ERRVEC+2
014014 012737 030114 000114             MOV     #MPERHD,@#MEMVEC ;SET MEM PARITY TRAP VECTOR
014022 012737 000340 000116             MOV     #PR7,@#MEMVEC+2 ;SET TRAP PRIORITY = 7
014030 004737 030030                    JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
014034 112737 000144 001115             MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
014042 105037 003120                    CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
014046 112737 000001 003136             MOVB    #1,HLPOVL       ;SET HLP FILE OVLD INDICTR
014054 104401 007024                    TYPE    ,DZR6N          ;TYPE PROGRAM I.D. FOR PART 2
014060 104401 007044                    TYPE    ,SUBVER
014064 104401 007125                    TYPE    ,PART2
014070 105737 003137                    TSTB    NOTYPE          ;SEE IF OPERATOR NOTE SHOULD BE TYPED
014074 001004                           BNE     39$             ;BR IF NOT
014076 104401 064640                    TYPE    ,NOTMSG         ;TYPE OPERATOR NOTE
014102 105237 003137                    INCB    NOTYPE          ;SET FLAG FOR NEXT TIME
014106 105737 003126            39$:    TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
014112 001402                           BEQ     40$             ;BR IF NOT
014114 104401 010417                    TYPE    ,DUACES         ;TYPE "* DUAL-ACCESS DATA TEST *"
014120 012737 027142 000060     40$:    MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
014126 012737 000200 000062             MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
014134 013701 003030                    MOV     RKVEC,R1        ;ADDR. OF RK06 VECTOR STORAGE
014140 012721 045620                    MOV     #I.INTR,(R1)+   ;SET IT TO RK06 HANDLER
014144 013711 003032                    MOV     RKPRI,(R1)      ;SET RK06 PRIORITY
014150 012737 027734 000250             MOV     #KTERHD,@#MMVEC ;VECT FOR KT11 FAILURE
014156 012737 000340 000252             MOV     #PR7,@#MMVEC+2  ;SET PRIOR. = 7 FOR HNDLER
014164 105037 003110                    CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
014170 012737 000000 177776             MOV     #PR0,@#PS       ;ALLOW ALL INTERRUPTS AGAIN
014176 012701 005222                    MOV     #PRVCMD,R1      ;ZERO OUT PREVIOUS COMMAND
014202 012700 000006                    MOV     #6,R0           ;SET COUNTER
014206 005021                   42$:    CLR     (R1)+           ;ZERO A WORD
014210 005300                           DEC     R0
014212 001375                           BNE     42$             ;BR IF NOT DONE YET
014214 005037 005502                    CLR     STALLS          ;DON'T ALLOW STALLS YET
014220 004737 027614                    JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
```

```
 4793  014224  012737  176543  053246  44$:    MOV     #176543,$HINUM  ;INIT. PSEUDO-RANDOM NOS.
 4794  014232  012737  123456  053250          MOV     #123456,$LONUM
 4795  014240  004737  027334                  JSR     PC,SIZMEM       ;SIZE MEMORY, FIX MA LIMIT IN PRM_IM
 4796  014244  105737  003125                  TSTB    XDPSVD          ;SEE IF XXDP PREVIOUSLY SAVED
 4797  014250  001404                          BEQ     9$              ;BR IF NOT
 4798  014252  004737  030536                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP
 4799  014256  105037  003125                  CLRB    XDPSVD          ;CLEAR THE FLAG
 4800  014262  105737  003106          9$:     TSTB    MDFLAG          ;SEE IF DEFAULT MODE
 4801  014266  001031                          BNE     10$             ;BR IF NOT DEFAULT MODE
 4802  014270  013700  001370                  MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
 4803  014274  013737  001374  003026          MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
 4804  014302  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
 4805  014310  001005                          BNE     18$             ;BR IF NO SIZING
 4806  014312  012700  120210                  MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
 4807  014316  012737  177440  003026          MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
 4808  014324  110037  003030          18$:    MOVB    R0,RKVEC        ;STORE VECTOR
 4809  014330  105037  003031                  CLRB    RKVEC+1         ;CLEAR HI BYTE
 4810  014334  000300                          SWAB    R0              ;GET PRTY INTO BITS 5-7
 4811  014336  042700  177437                  BIC     #177437,R0      ;CLEAR OTHER BITS
 4812  014342  010037  003032                  MOV     R0,RKPRI        ;STORE RK06 PRIORITY
 4813  014346  000137  015250                  JMP     ALLDRV          ;GO CHECK ALL DRIVES
 4814
 4815
 4816
 4817                                  ;BEGIN PARAMETER INPUT MODE
 4818  014352  104401  007216          10$:    TYPE    ,PRMINP         ;TYPE "PARAMETER INPUT MODE"
 4819
 4820                                  ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
 4821  014356  013746  003026                  MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
 4822  014362  104401  007250                  TYPE    ,RKBADR         ;TYPE "RK06 BUS ADR = "
 4823  014366  004737  027504                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
 4824  014372  012637  003026                  MOV     (SP)+,RKBAS     ;STORE NEW VALUE
 4825                                  ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
 4826  014376  013746  003030                  MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
 4827  014402  104401  007270                  TYPE    ,RKVADR         ;TYPE "RK06 VEC ADR = "
 4828  014406  004737  027504                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
 4829  014412  012637  003030                  MOV     (SP)+,RKVEC     ;STORE NEW VALUE
 4830                                  ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
 4831  014416  013746  003032          11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
 4832  014422  006316                          ASL     (SP)            ;GET IT INTO BITS 0-2
 4833  014424  006316                          ASL     (SP)
 4834  014426  006316                          ASL     (SP)
 4835  014430  000316                          SWAB    (SP)
 4836  014432  104401  007310                  TYPE    ,RkPRTY         ;TYPE   "RK06 PRIORITY = "
 4837  014436  004737  027504                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
 4838  014442  012600                          MOV     (SP)+,R0
 4839  014444  020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
 4840  014450  002414                          BLT     12$             ;BR IF NEW VALUE TOO SMALL
 4841  014452  020027  000007                  CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
 4842  014456  003011                          BGT     12$             ;BR IF NEW VALUE TOO LARGE
 4843  014460  000300                          SWAB    R0              ;GET PRIORITY INTO BITS 5-7
 4844  014462  006200                          ASR     R0
 4845  014464  006200                          ASR     R0
 4846  014466  006200                          ASR     R0
 4847  014470  010037  003032                  MOV     R0,RKPRI        ;STORE NEW VALUE
 4848  014474  104401  001315                  TYPE    ,$CRLF
```

```
 4849  014500  000405                        BR      16$
 4850  014502  104401  005262          12$:  TYPE    .BUFFO          ;ECHO BAD INPUT
 4851  014506  104401  001314                TYPE    .$QUES
 4852  014512  000741                        BR      11$             ;GO ASK AGAIN
 4853
 4854  014514  105037  003116          16$:  CLRB    ERRCNT          ;CLEAR ERROR CNT FOR RESTARTS
 4855  014520  012737  014532  000042        MOV     #DRVTST,@#42    ;SET UP DUMP MODE RETURN FROM $EOP
 4856                                                                ;    UPON COMPLETION OF REQUESTED PASSES
 4857  014526  000137  015342                JMP     CHKLST          ;GO CHECK STATUS OF MARKED DRIVES
 4858
 4859
 4860
 4861                                ;;*****************************************************************
 4862                                ;SBTTL  TO - DESIRED DRIVE INPUT ROUTINE
 4863                                ;*THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
 4864                                ;*WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
 4865                                ;*CHOICE OF DRIVES IS REQUESTED BY TTY INPUT. AND THESE
 4866                                ;*DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
 4867                                ;*AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
 4868                                ;*LIST (DRVLST).
 4869                                ;;*****************************************************************
 4870
 4871  014532                        DRVTST:
 4872  014532  012706  001100                MOV     #STACK,SP       ;RESET THE STACK
 4873  014536  005037  005502                CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
 4874  014542  004737  030030                JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
 4875  014546  104401  007724                TYPE    ,ALDRVS         ;ASK IF ALL DRIVES DESIRED
 4876  014552  004737  032272                JSR     PC,RDCHRS       ;READ RESPONSE
 4877  014556  014532                        DRVTST                  ;(↑C) RETURN ADDRESS
 4878  014560  014532                        DRVTST                  ;(↑Z) RETURN ADDRESS
 4879  014562  014532                        DRVTST                  ;(↑U) OR ERROR RETURN ADDRESS
 4880  014564  005700                        TST     R0              ;SEE IF NULL INPUT
 4881  014566  001413                        BEQ     TELDRV          ;BR IF NULL INPUT
 4882  014570  022737  000101  005262        CMP     #'A,BUFFO       ;SEE IF ALL DRIVES REQUESTED
 4883  014576  001002                        BNE     4$              ;BR IF NOT ALL DRIVES
 4884  014600  000137  015250                JMP     ALLDRV          ;CHECK ALL DRIVES
 4885  014604  104401  005262          4$:   TYPE    .BUFFO          ;ECHO BAD INPUT
 4886  014610  104401  001314                TYPE    .$QUES          ;TYPE <?> AND <CR>,<LF>
 4887  014614  000746                        BR      DRVTST          ;GO ASK AGAIN
 4888                                ;TYPE CURRENT DRIVE LIST
 4889  014616  104401  007350          TELDRV: TYPE    ,DRVSEQ         ;TYPE "DRIVE(S) = "
 4890  014622  005001                        CLR     R1              ;INITIALIZE DRIVE NUMBER
 4891  014624  005000                        CLR     R0              ;INIT. COUNT OF LISTED DRIVES
 4892  014626  012703  000001                MOV     #BIT0,R3        ;INIT BIT POINTER
 4893  014632  105761  005606          4$:   TSTB    DRVLST(R1)      ;SEE IF THIS DRIVE IS LISTED
 4894  014636  001414                        BEQ     8$              ;BR IF DRIVE NOT LISTED
 4895  014640  005700                        TST     R0              ;SEE IF FIRST TIME HERE
 4896  014642  001402                        BEQ     6$              ;BR IF FIRST TIME HERE
 4897  014644  104401  013417                TYPE    ,COMMA          ;TYPE A COMMA
 4898  014650  010137  005532          6$:   MOV     R1,SCRACH       ;USE SCRACH FOR BUFFER
 4899  014654  052737  000060  005532        BIS     #'0,SCRACH      ;CONVERT DRIVE NO. TO ASCII
 4900  014662  104401  005532                TYPE    ,SCRACH         ;TYPE DRIVE NO.
 4901  014666  005200                        INC     R0              ;INCREMENT NO. OF LISTED DRIVES
 4902  014670  005201                  8$:   INC     R1              ;INCREMENT DRIVE NO.
 4903  014672  006303                        ASL     R3              ;SHIFT BIT POINTER
 4904  014674  022701  000010                CMP     #10,R1          ;SEE IF DONE YET
```

```
4905  014700  001354                        BNE     4$              ;BR IF NOT DONE
4906  014702  005700                        TST     R0              ;SEE IF ANY DRIVES WERE LISTED
4907  014704  001016                        BNE     26$             ;BR IF YES
4908  014706  104401  011271                TYPE    ,NOFALS         ;TYPE "  NONE"
4909  014712  104401  007571                TYPE    ,NODRTS         ;TYPE "** NO DRIVES TO TEST"
4910                                                                ;     "PRESS 'CONT' WHEN RDY"
4911  014716  012703  000401                MOV     #401,R3         ;PATTERN TO MARK DRIVES
4912  014722  012701  005606                MOV     #DRVLST,R1      ;DRIVE LIST ADDRESS
4913  014726  010321                        MOV     R3,(R1)+        ;MARK ALL DRIVES IN LIST
4914  014730  010321                        MOV     R3,(R1)+
4915  014732  010321                        MOV     R3,(R1)+
4916  014734  010311                        MOV     R3,(R1)
4917  014736  000137  044476                JMP     HLTPRG          ;HALT PROGRAM
4918  014742  104401  001315        26$:    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
4919  014746  105737  003106                TSTB    MDFLAG          ;SEE IF DEFAULT MODE
4920  014752  001002                        BNE     20$             ;BR IF NOT DEFAULT MODE
4921  014754  000137  015674                JMP     LODFLS          ;GO LOAD DEFLT ITER. COUNTS
4922  014760  122737  000013  000041  20$:  CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
4923  014766  001032                        BNE     19$             ;BR IF NOT
4924  014770  105737  005606                TSTB    DRVLST          ;SEE IF DRIVE 0 LISTED TO TEST
4925  014774  001427                        BEQ     19$             ;BR IF NOT
4926  014776  104401  007502        24$:    TYPE    ,REPLPK         ;TYPE "IF XXDP PACK ON DRV 0, TYPE Y <CR>
4927                                                                ;     AND REPLACE IT"
4928  015002  004737  032272                JSR     PC,RDCHRS       ;READ RESPONSE
4929  015006  014776                        24$                     ;(!C) RETURN
4930  015010  014776                        24$                     ;(!Z) RETURN
4931  015012  014776                        24$                     ;(!U) RETURN
4932  015014  005700                        TST     R0              ;SEE IF NULL INPUT
4933  015016  001416                        BEQ     19$             ;BR IF JUST <CR> TYPED
4934  015020  022737  000131  005262        CMP     #'Y,BUFF0       ;SEE IF "Y <CR>" TYPED
4935  015026  001363                        BNE     24$             ;BR IF NOT, TO ASK AGAIN
4936  015030  104401  007622                TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN DRV RDY"
4937  015034  042762  000100  000000        BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
4938  015042  000000                        HALT                    ;HALT FOR PACK CHANGE
4939  015044  004737  030776                JSR     PC,INITSS       ;INIT THE SUB-SYS
4940  015050  000137  015342                JMP     CHKLST          ;GO CHECK STATUS OF LISTED DRIVES
4941  015054  104401  013446        19$:    TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
4942                                ;READ AND CHECK NEW DRIVE NUMBERS
4943  015060  004737  032272                JSR     PC,RDCHRS       ;READ IN INPUT STRING
4944  015064  014532                        DRVTST                  ;(!C) RETURN ADDRESS
4945  015066  014532                        DRVTST                  ;(!Z) RETURN ADDRESS
4946  015070  014616                        TELDRV                  ;(!U) OR ERROR RETURN ADDRESS
4947  015072  005700                        TST     R0              ;SEE IF NULL INPUT
4948  015074  001007                        BNE     9$              ;BR IF NEW DRIVES WILL BE SELECTED
4949  015076  105737  003126                TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
4950  015102  001002                        BNE     22$             ;BR IF YES
4951  015104  000137  015406                JMP     ASKTST          ;JUMP TO THE TEST INPUT ROUTINE
4952  015110  000137  016152        22$:    JMP     INPUTP          ;GO ASK FOR INPUT PARAMETERS
4953  015114  022700  000017        9$:     CMP     #15.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
4954  015120  002005                        BGE     12$             ;BR IF NOT TOO MANY
4955  015122  104401  005262        10$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
4956  015126  104401  001314                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
4957  015132  000631                        BR      TELDRV          ;GO TYPE DRIVES AND ASK AGAIN
4958  015134  005001                12$:    CLR     R1              ;CLEAR CHAR. POINTER
4959  015136  126127  005262  000060  14$:  CMPB    BUFF0(R1),#'0   ;SEE IF DRIVE NO. < 0
4960  015144  002766                        BLT     10$             ;BR TO ECHO BAD INPUT
```

```
 4961  015146  126127  005262  000067          CMPB    BUFFO(R1),#'7   ;SEE IF > 7
 4962  015154  003362                          BGT     10$             ;BR TO ECHO BAD INPUT
 4963  015156  005201                          INC     R1              ;INCREMENT CHAR POINTER
 4964  015160  020100                          CMP     R1,R0           ;SEE IF MORE CHARS TO CHECK
 4965  015162  001406                          BEQ     16$             ;BR IF ALL CHARS CHECKED
 4966  015164  126127  005262  000054          CMPB    BUFFO(R1),#',   ;SEE IF THIS IS COMMA
 4967  015172  001353                          BNE     10$             ;BR IF NOT COMMA
 4968  015174  005201                          INC     R1              ;INCREMENT CHAR POINTER
 4969  015176  000757                          BR      14$             ;BR TO CONTINUE CHECKING CHARS
 4970                          ;GET NEW DRIVE NUMBERS INTO LIST
 4971  015200  012701  005606          16$:    MOV     #DRVLST,R1      ;GET DRIVE LIST ADDRESS
 4972  015204  005021                          CLR     (R1)+           ;CLEAR OUT THE DRIVE LIST
 4973  015206  005021                          CLR     (R1)+
 4974  015210  005021                          CLR     (R1)+
 4975  015212  005011                          CLR     (R1)
 4976  015214  005000                          CLR     R0              ;CLEAR BUFFER POINTER
 4977  015216  116001  005262          18$:    MOVB    BUFFO(R0),R1    ;GET A DRIVE NUMBER
 4978  015222  042701  000060                  BIC     #'0,R1          ;STRIP ASCII BITS
 4979  015226  112761  000001  005606          MOVB    #1,DRVLST(R1)   ;MARK THIS DRIVE IN DRIVE LIST
 4980  015234  062700  000002                  ADD     #2,R0           ;POINT TO NEXT DRIVE NUMBER
 4981  015240  105760  005261                  TSTB    BUFFO-1(R0)     ;SEE IF NULL CHAR YET
 4982  015244  001364                          BNE     18$             ;BR IF NOT DONE YET
 4983  015246  000435                          BR      CHKLST          ;GO CHECK NEW DRIVES FOR VALID STATUS
 4984                          ;COME HERE IF ALL DRIVES REQUESTED
 4985  015250  005000          ALLDRV: CLR     R0              ;CLEAR DRIVE NUMBER
 4986  015252  013703  001376                  MOV     $DEVM,R3        ;GET APT DEVICE MAP
 4987  015256  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
 4988  015264  001002                          BNE     1$              ;BR IF NO SIZING
 4989  015266  012703  000377                  MOV     #377,R3         ;SET UP FOR SIZING
 4990  015272  012701  000001          1$:     MOV     #BIT0,R1        ;SET BIT POINTER
 4991  015276  105060  005606          2$:     CLRB    DRVLST(R0)      ;INITIALIZE DRIVE ENTRY
 4992  015302  030103                          BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
 4993  015304  001403                          BEQ     4$              ;BR IF NOT
 4994  015306  112760  000001  005606          MOVB    #1,DRVLST(R0)   ;MARK THIS DRIVE IN DRIVE LIST
 4995  015314  006301                  4$:     ASL     R1              ;SHIFT BIT POINTER
 4996  015316  005200                          INC     R0              ;INCREMENT DRIVE NUMBER
 4997  015320  022700  000010                  CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
 4998  015324  001364                          BNE     2$              ;BR IF NOT DONE YET
 4999  015326  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF PROGRAM SHOULD SIZE
 5000  015334  001402                          BEQ     CHKLST          ;BR IF YES
 5001  015336  000137  014616                  JMP     TELDRV          ;GO LIST APT-SELECTED DRIVES
 5002                          ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
 5003  015342  005000          CHKLST: CLR     R0              ;CLEAR DRIVE NUMBER
 5004  015344  105760  005606          2$:     TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
 5005  015350  001405                          BEQ     4$              ;BR IF NOT MARKED
 5006  015352  010037  005500                  MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
 5007  015356  004737  031116                  JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE
 5008                                                                  ;IF NOT VALID, TYPE MESSAGE
 5009                                                                  ;   AND REMOVE IT FROM DRIVE LIST
 5010  015362  015400                          6$                      ;ERROR RETURN ADDRESS FOR SCNDRV
 5011  015364  005200                  4$:     INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
 5012  015366  022700  000010                  CMP     #10,R0          ;SEE IF DONE CHECKING LIST
 5013  015372  001364                          BNE     2$              ;BR IF NOT DONE YET
 5014  015374  000137  014616                  JMP     TELDRV          ;GO BACK TO LIST SELECTED DRIVES
 5015  015400  105060  005606          6$:     CLRB    DRVLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
 5016  015404  000767                          BR      4$              ;CONTINUE CHECKING THE LIST
```

```
                                        ;;*****************************************************************
                                        .SBTTL  TO - DESIRED TEST INPUT ROUTINE
                                        ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
                                        ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
                                        ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
                                        ;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
                                        ;*THAT THE TEST IS NOT TO BE RUN.
                                        ;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
                                        ;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
                                        ;*TYPED, (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
                                        ;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
                                        ;*OF OPERATING PARAMETERS, AND RUN TESTS.
                                        ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
                                        ;* (↑Z) CAUSES RETURN TO ASKTMD.
                                        ;;*****************************************************************

                                        ;DETERMINE L,C, OR I MODE
015406  104401  010007          ASKTST: TYPE    ,TSTMDS         ;ASK FOR TEST INPUT MODE
015412  104401  010111          ASKTMD: TYPE    ,ENTLCI         ;TYPE "ENTER L,C, OR I"
015416  004737  032272                  JSR     PC,RDCHRS       ;READ RESPONSE
015422  014532                          DRVTST                  ;(↑C) RETURN ADDRESS
015424  015412                          ASKTMD                  ;(↑Z) RETURN ADDRESS
015426  015412                          ASKTMD                  ;(↑U) OR ERROR RETURN ADDRESS
015430  005700                          TST     R0              ;SEE IF ANY INPUT
015432  001005                          BNE     4$              ;BR IF ANY INPUT
015434  104401  005262          2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
015440  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
015444  000762                          BR      ASKTMD          ;GO ASK AGAIN
015446  022737  000114  005262  4$:     CMP     #'L,BUFFO       ;SEE IF (L) TYPED
015454  001002                          BNE     6$              ;BR IF NOT (L)
015456  000137  015512                  JMP     LSTTST          ;JUMP TO LIST TESTS
015462  022737  000103  005262  6$:     CMP     #'C,BUFFO       ;SEE IF (C) TYPED
015470  001002                          BNE     8$              ;BR IF NOT (C)
015472  000137  015630                  JMP     CHGTST          ;JUMP TO CHANGE TESTS
015476  022737  000111  005262  8$:     CMP     #'I,BUFFO       ;SEE IF (I) TYPED
015504  001353                          BNE     2$              ;BR IF NOT (I), TO ECHO BAD INPUT
015506  000137  016152                  JMP     INPUTP          ;GO INPUT PARAMS AND RUN TESTS

                                        ;LIST CURRENT TESTS AND ITERATION COUNTS
015512  104401  010213          LSTTST: TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
015516  005001                          CLR     R1              ;INITIALIZE TEST INDEX
015520  004737  027302          2$:     JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
015524  004737  032604                  JSR     PC,TYPTST       ;TYPE CURRENT TEST AND ITERATION NUMBER
015530  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
015534  005737  005522                  TST     INTCHR          ;SEE IF ANY INPUT
015540  001416                          BEQ     8$              ;BR IF NO INPUT
015542  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
015550  001002                          BNE     4$              ;BR IF NOT (↑C)
015552  000137  014532                  JMP     DRVTST          ;JUMP TO ASK FOR DRIVES AGAIN
015556  122737  000032  005522  4$:     CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
015564  001002                          BNE     6$              ;BR IF NOT (↑Z)
015566  000137  015412                  JMP     ASKTMD          ;JUMP TO ASK FOR NEW TEST INPUT MODE
015572  004737  027322          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
015576  062701  000002          8$:     ADD     #2,R1           ;INCREMENT TEST INDEX
```

```
5073  015602  010102                        MOV     R1,R2           ;GET COPY OF INDEX
5074  015604  062702  005616                ADD     #TSTLST,R2      ;GET POSITION IN LIST
5075  015610  022702  005632                CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
5076  015614  001341                        BNE     2$              ;BR IF NOT DONE YET
5077  015616  042777  000100  163320        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
5078  015624  000137  015412                JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
5079
5080                          ;CHANGE CURRENT TESTS AND ITERATION COUNTS
5081  015630  104401  010137  CHGTST: TYPE    ,DFTEST         ;ASK IF TESTS SHOULD BE DEFAULTED
5082  015634  004737  032272                JSR     PC,RDCHRS       ;READ RESPONSE
5083  015640  014532                        DRVTST                  ;(↑C) RETURN ADDRESS
5084  015642  015412                        ASKTMD                  ;(↑Z) RETURN ADDRESS
5085  015644  015630                        CHGTST                  ;(↑U) OR ERROR RETURN ADDRESS
5086  015646  005700                        TST     R0              ;SEE IF NULL INPUT
5087  015650  001426                        BEQ     NULINP          ;BR IF NULL INPUT
5088  015652  022737  000104  005262        CMP     #'D,BUFFO       ;SEE IF (D) TYPED
5089  015660  001405                        BEQ     LODFLS          ;BR IF DEFAULTS REQUESTED
5090  015662  104401  005262                TYPE    ,BUFFO          ;ECHO BAD INPUT
5091  015666  104401  001314                TYPE    ,SQUES          ;TYPE <?> AND <CR>,<LF>
5092  015672  000756                        BR      CHGTST          ;GO ASK AGAIN
5093                          ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
5094  015674  012700  005632  LODFLS: MOV     #DFLTST,R0      ;LOAD DEFAULT LIST ADDRESS
5095  015700  012702  005616                MOV     #TSTLST,R2      ;LOAD TEST LIST ADDRESS
5096  015704  012022          4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
5097  015706  022702  005632                CMP     #DFLTST,R2      ;SEE IF DONE YET
5098  015712  001374                        BNE     4$              ;BR IF NOT DONE YET
5099  015714  105737  003106                TSTB    MDFLAG          ;SEE IF DEFAULT MODE
5100  015720  001234                        BNE     ASKTMD          ;BR IF NOT DEFAULT MODE
5101  015722  000137  016706                JMP     LODFPT          ;GO LOAD DEFAULT PARAMETERS
5102  015726  005001  NULINP: CLR     R1              ;INITIALIZE TEST INDEX
5103  015730  104401  010213                TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
5104                          ;TYPE CURRENT TEST AND ITERATION NUMBER
5105  015734  004737  032604  8$:     JSR     PC,TYPTST       ;TYPE A TEST NO. AND ITERATION NO.
5106  015740  104401  013426                TYPE    ,SPACE1         ;TYPE A SPACE
5107  015744  104401  013446                TYPE    ,PRCMPT         ;TYPE ASTERISK AND SPACE
5108                          ;READ AND CHECK INPUT, IF ANY
5109  015750  004737  032272                JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
5110  015754  014532                        DRVTST                  ;(↑C) RETURN ADDRESS
5111  015756  015412                        ASKTMD                  ;(↑Z) RETURN ADDRESS
5112  015760  015734                        8$                      ;(↑U) OR ERROR RETURN ADDRESS
5113  015762  005700                        TST     R0              ;SEE IF ANY INPUT
5114  015764  001012                        BNE     12$             ;BR IF ANY INPUT
5115  015766  062701  000002  10$:     ADD     #2,R1           ;INCREMENT INDEX
5116  015772  010102                        MOV     R1,R2           ;COPY INDEX
5117  015774  062702  005616                ADD     #TSTLST,R2      ;GET POSITION IN LIST
5118  016000  022702  005632                CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
5119  016004  001353                        BNE     8$              ;BR IF NOT DONE YET
5120  016006  000137  015412                JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
5121  016012  022737  000041  005262  12$:     CMP     #'!,BUFFO       ;SEE IF (!) TYPED
5122  016020  001431                        BEQ     16$             ;BR TO PROPAGATE CURRENT ITER. NO.
5123  016022  005002                        CLR     R2              ;INITIALIZE (!) INDICATOR
5124  016024  122760  000041  005261        CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
5125  016032  001004                        BNE     14$             ;BR IF NOT (!)
5126  016034  105060  005261                CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
5127  016040  005300                        DEC     R0              ;DECREMENT CHAR COUNT
5128  016042  005202                        INC     R2              ;SET (!) INDICATOR
```

```
5129  016044  022700  000005          14$:    CMP     #5,R0           ;SEE HOW MANY CHARS NOW
5130  016050  002433                          BLT     20$             ;BR IF TOO MANY (MAX ITER. NO. = 77775,
5131  016052  012746  005262                  MOV     #BUFFO,-(SP)    ;GET BUF ADDR. ON STACK FOR OCTBIN
5132  016056  004737  053014                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
5133  016062  016140                          20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
5134  016064  012600                          MOV     (SP)+,R0        ;GET ITERATION NUMBER
5135  016066  020027  077776                  CMP     R0,#77776       ;SEE IF > 77776
5136  016072  101022                          BHI     20$             ;BR IF TOO BIG
5137  016074  010061  005616                  MOV     R0,TSTLST(R1)   ;PUT ITERATION NUMBER INTO TEST LIST
5138  016100  005702                          TST     R2              ;SEE IF (!) WAS TYPED
5139  016102  001731                          BEQ     10$             ;BR IF NOT (!)
5140                                  ;PROPAGATE CURRENT ITERATION NO. TO THE END OF TSTLST
5141  016104  010102                  16$:    MOV     R1,R2           ;COPY INDEX
5142  016106  062702  005616                  ADD     #TSTLST,R2      ;GET POSITION IN LIST
5143  016112  022702  005630                  CMP     #DFLTST-2,R2    ;SEE IF AT END OF LIST
5144  016116  001002                          BNE     17$             ;BR IF NOT DONE
5145  016120  000137  015412                  JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
5146  016124  016161  005616  005620  17$:    MOV     TSTLST(R1),TSTLST+2(R1) ;PROPAGATE TO NEXT WORD
5147  016132  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
5148  016136  000762                          BR      16$             ;BR TO CONTINUE
5149  016140  104401  005262          20$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
5150  016144  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5151  016150  000671                          BR      8$              ;GO ASK AGAIN
5152
5153                                  ;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
5154  016152  104401  010243          INPUTP: TYPE    ,EXPLAN         ;EXPLAIN INPUT MODES
5155  016156  005037  005502          ASKMDE: CLR     $TALLS          ;INHIBIT STALL BETWEEN OPERATIONS
5156  016162  004737  030030                  JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
5157  016166  104401  010371                  TYPE    ,PARMDE         ;ASK FOR DESIRED INPUT MODE
5158  016172  104401  013446                  TYPE    ,PROMPT         ;TYPE "* "
5159  016176  004737  032272                  JSR     PC,RDCHRS       ;READ RESPONSE TO MODE QUESTION
5160  016202  014532                          DRVTST                  ;(↑C) RETURN ADDRESS
5161  016204  016156                          ASKMDE                  ;(↑Z) RETURN ADDRESS
5162  016206  016156                          ASKMDE                  ;(↑U) OR ERROR RETURN ADDRESS
5163  016210  005700                          TST     R0              ;SEE IF NULL INPUT
5164  016212  001005                          BNE     4$              ;BR IF ANY INPUT
5165  016214  104401  005262          2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
5166  016220  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5167  016224  000754                          BR      ASKMDE          ;GO ASK AGAIN
5168  016226  022737  000124  005262  4$:     CMP     #'T,BUFFO       ;SEE IF (T) TYPED
5169  016234  001002                          BNE     6$              ;BR IF NOT (T)
5170  016236  000137  016370                  JMP     TYPLST          ;JUMP TO THE TYPE LIST ROUTINE
5171  016242  022737  000117  005262  6$:     CMP     #'O,BUFFO       ;SEE IF (O) TYPED
5172  016250  001002                          BNE     8$              ;BR IF NOT (O)
5173  016252  000137  016642                  JMP     OPNLST          ;JUMP TO THE OPEN LIST ROUTINE
5174  016256  022737  000123  005262  8$:     CMP     #'S,BUFFO       ;SEE IF (S) TYPED
5175  016264  001002                          BNE     10$             ;BR IF NOT (S)
5176  016266  000137  017126                  JMP     SETPRM          ;JUMP TO THE SET INDIV. PARAM. ROUTINE
5177  016272  022737  000122  005262  10$:    CMP     #'R,BUFFO       ;SEE IF (R) TYPED
5178  016300  001345                          BNE     2$              ;BR IF NOT (R), TO ECHO BAD INPUT
5179  016302  023737  005656  005660          CMP     S0,S1           ;COMPARE 20(DEC) SECTOR LIMITS
5180  016310  003403                          BLE     12$             ;BR IF S0 NOT > S1
5181  016312  104401  010477                  TYPE    ,SECNL1         ;TYPE "S0>S1 NOT ALLOWED"
5182  016316  000717                          BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
5183  016320  023737  005662  005664  12$:    CMP     S2,S3           ;COMPARE 22(DEC) SECTOR LIMITS
5184  016326  003405                          BLE     14$             ;BR IF S2 NOT > S3
```

```
 5185  016330  104401  010527                    TYPE    .SECNL2         ;TYPE "S2,S3"
 5186  016334  104401  010510                    TYPE    .NOTALD         ;TYPE "NOT ALLOWED"
 5187  016340  000706                            BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
 5188  016342  023737  005652  005654  14$:     CMP     FT,LT           ;COMPARE CHOSEN TRACK LIMITS
 5189  016350  003405                            BLE     20$             ;BR IF FT NOT > LT
 5190  016352  104401  010541                    TYPE    .TRKNLW         ;TYPE "FT>LT"
 5191  016356  104401  010510                    TYPE    .NOTALD         ;TYPE "NOT ALLOWED"
 5192  016362  000675                            BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
 5193  016364  000137  017362          20$:     JMP     RUNTST          ;GO RUN THE TESTS
 5194
 5195
 5196
 5197                                    .SBTTL  TO - TYPE (T) LIST ROUTINE
 5198                                    ;*THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
 5199                                    ;*CONSOLE. LINE BY LINE. EACH LINE HAS THE FOLLOWING
 5200                                    ;*FORMAT:  XX=YYYYYY  , WHERE XX IS A PARAMETER MNEMONIC
 5201                                    ;*AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
 5202                                    ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST.
 5203                                    ;*AND (↑Z) CAUSES RETURN TO ASKMDE.
 5204
 5205  016370                            TYPLST:
 5206  016370  005001                            CLR     R1              ;INITIALIZE INDEX
 5207  016372  004737  027302                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 5208  016376  004737  032640          1$:      JSR     PC,TYPPRM       ;TYPE CURRENT PARAMETER AND VALUE
 5209  016402  104401  001315                    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 5210  016406  005737  005522                    TST     INTCHR          ;SEE IF ANY INPUT AT KBD
 5211  016412  001420                            BEQ     8$              ;BR IF NO INPUT
 5212  016414  122737  000003  005522            CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
 5213  016422  001002                            BNE     4$              ;BR IF NOT (↑C)
 5214  016424  000137  014532                    JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
 5215  016430  122737  000032  005522  4$:      CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
 5216  016436  001002                            BNE     6$              ;BR IF NOT (↑Z)
 5217  016440  000137  016156                    JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5218  016444  004737  027322          6$:      JSR     PC,ECOBAD       ;ECHO BAD INPUT
 5219  016450  004737  027302                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 5220  016454  022761  040515  005776  8$:      CMP     #"MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
 5221  016462  001002                            BNE     10$             ;BR IF NOT (MA)
 5222  016464  062701  000002                    ADD     #2,R1           ;INCREMENT PARAMETER INDEX
 5223  016470  062701  000002          10$:     ADD     #2,R1           ;INCREMENT PARAMETER INDEX
 5224  016474  010102                            MOV     R1,R2           ;GET COPY OF INDEX
 5225  016476  062702  005646                    ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
 5226  016502  022702  005674                    CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
 5227  016506  001333                            BNE     1$              ;BR IF NOT DONE YET
 5228  016510  032737  100000  005666            BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
 5229  016516  001005                            BNE     12$             ;BR IF PATTERN SPECIFIED
 5230  016520  042777  000100  162416            BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 5231  016526  000137  016156                    JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5232                                    ;TYPE OUT USER-DEFINED PATTERN 15
 5233  016532  104401  010666          12$:     TYPE    ,PFIFTN         ;TYPE "USER-DEFINED PATTERN 15 :"
 5234  016536  005001                            CLR     R1              ;INITIALIZE WORD INDEX
 5235  016540  005737  005522          14$:     TST     INTCHR          ;SEE IF ANY INPUT
 5236  016544  001420                            BEQ     20$             ;BR IF NO INPUT
 5237  016546  122737  000003  005522            CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
 5238  016554  001002                            BNE     16$             ;BR IF NOT (↑C)
 5239  016556  000137  014532                    JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
 5240  016562  122737  000032  005522  16$:     CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
```

```
5241  016570  001002                        BNE     18$             ;BR IF NOT (↑Z)
5242  016572  000137  016156                JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5243  016576  004737  027322        18$:    JSR     PC,ECOBAD       ;ECHO BAD INPUT
5244  016602  004737  027302                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5245  016606  004737  032720        20$:    JSR     PC,TYPPAT       ;TYPE WORD XX = YYYYYY
5246  016612  104401  001315                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5247  016616  062701  000002                ADD     #2,R1           ;INCREMENT INDEX
5248  016622  022701  000040                CMP     #32.,R1         ;SEE IF 16 WORDS TYPED
5249  016626  001344                        BNE     14$             ;BR IF NOT DONE YET
5250  016630  042777  000100  162306        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
5251  016636  000137  016156                JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5252
5253
5254
5255                                  .SBTTL  TO - OPEN (O) LIST ROUTINE
5256                                  ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
5257                                  ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
5258                                  ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
5259                                  ;*TTY INPUT. TYPING (↑C) CAUSES IMMEDIATE RETURN TO
5260                                  ;*DRVTST, AND (↑Z) CAUSES RETURN TO ASKMDE.
5261
5262
5263  016642                          OPNLST:
5264  016642  104401  010604                TYPE    ,DFQUES         ;ASK IF ALL DEFAULT VALUES DESIRED
5265  016646  004737  032272                JSR     PC,RDCHRS       ;READ RESPONSE TO DEFAULT QUESTION
5266  016652  014532                        DRVTST                  ;(↑C) RETURN ADDRESS
5267  016654  016156                        ASKMDE                  ;(↑Z) RETURN ADDRESS
5268  016656  016642                        OPNLST                  ;(↑U) OR ERROR RETURN ADDRESS
5269  016660  005700                        TST     R0              ;SEE IF NULL INPUT
5270  016662  001433                        BEQ     NOTDFT          ;BR IF DEFAULTS NOT REQUESTED
5271  016664  022737  000104  005262        CMP     #'D,BUFFO       ;SEE IF (D) TYPED
5272  016672  001405                        BEQ     LODFPT          ;BR IF DEFAULTS DESIRED
5273  016674  104401  005262                TYPE    ,BUFFO          ;ECHO BAD INPUT
5274  016700  104401  001314                TYPE    ,$QUES
5275  016704  000756                        BR      OPNLST          ;GO ASK AGAIN
5276                                  ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
5277  016706  012700  005674        LODFPT: MOV     #PRDFLT,R0      ;LOAD DEFAULT LIST ADDRESS
5278  016712  012702  005646                MOV     #PRMLST,R2      ;LOAD PARAMETER LIST ADDRESS
5279  016716  012022                4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
5280  016720  022702  005674                CMP     #PRDFLT,R2      ;SEE IF DONE YET
5281  016724  001374                        BNE     4$              ;BR IF MORE VALUES TO LOAD YET
5282  016726  105737  003106                TSTB    MDFLAG          ;SEE IF DEFAULT MODE
5283  016732  001005                        BNE     6$              ;BR IF NOT DEFAULT MODE
5284  016734  012737  000001  023540        MOV     #1,$EOPCT       ;SET PASS COUNT = 1
5285  016742  000137  017444                JMP     STPASS          ;GO RUN DFLT TESTS
5286  016746  000137  016156        6$:     JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5287  016752  005001                NOTDFT: CLR     R1              ;INITIALIZE INDEX
5288                                  ;TYPE CURRENT PARAMETER AND VALUE
5289  016754  004737  032640        8$:     JSR     PC,TYPPRM       ;TYPE PARAMETER AND VALUE
5290  016760  104401  013426                TYPE    ,SPACE1         ;TYPE A SPACE
5291  016764  104401  013446                TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
5292                                  ;READ AND CHECK INPUT, IF ANY
5293  016770  004737  032272                JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
5294  016774  014532                        DRVTST                  ;(↑C) RETURN ADDRESS FOR RDCHRS
5295  016776  016156                        ASKMDE                  ;(↑Z) RETURN ADDRESS FOR RDCHRS
5296  017000  016754                        8$                      ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
```

```
005700                          TST     R0              ;SEE IF ANY INPUT
001007                          BNE     10$             ;BR IF ANY INPUT
022761  040515  005776          CMP     #^MA,PRMNEM(R1) ;SEE IF (MA) JUST DEFAULTED
001023                          BNE     12$             ;BR IF NOT (MA)
062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
000420                          BR      12$             ;BR TO MOVE ON TO NEXT PARAMETER
022700  000010          10$:    CMP     #10,R0          ;SEE IF 8 CHARACTERS TYPED
002431                          BLT     14$             ;BR IF MORE THAN 8 TYPED
012746  005262                  MOV     #BUFFO,-(SP)    ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
004737  053014                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
017114                          14$                     ;ERROR RETURN ADDRESS FOR OCTBIN
012637  005466                  MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
013737  053146  005470          MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
                        ;CHECK PARAMETER VALUE AND PUT INTO LIST
004737  033276                  JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAM VALUE
017114                          14$                     ;ERROR RETURN ADDR. FOR CHKPRM
                        ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
004737  032752          12$:    JSR     PC,MODP15
                        ;MOVE ON TO NEXT PARAMETER
062701  000002                  ADD     #2,R1           ;INCREMENT THE PARAMETER INDEX
010102                          MOV     R1,R2           ;GET COPY OF INDEX
062702  005646                  ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
022702  005674                  CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
001322                          BNE     8$              ;BR IF NOT DONE YET
000137  016156                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
104401  005262          14$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
000713                          BR      8$              ;BR TO ASK AGAIN
```

```
                        .SBTTL  TO - SET (S) INDIVIDUAL PARAM ROUTINE
                        ;*THIS ROUTINE IS USED TO ALLOW SETTING OF ANY INDIVIDUAL
                        ;*PARAMETER VALUE BY TTY INPUT. A PROMPTER (>) IS
                        ;*TYPED AND THE ROUTINE WAITS FOR INPUT OF THE FORM
                        ;* XX=YYYYYY, WHERE XX IS THE MNEMONIC FOR ANY VALID
                        ;*PARAMETER IN THE LIST, AND YYYYYY IS THE DESIRED PARAMETER
                        ;*VALUE IN OCTAL DIGITS. THE PROGRAM THEN ASKS FOR ANOTHER
                        ;*PARAMETER BY TYPING "> " AGAIN. TYPING (↑C) CAUSES
                        ;*IMMEDIATE RETURN TO DRVTST, AND (↑Z) CAUSES RETURN TO
                        ;*ASKMDE.

017126                  SETPRM:
104401  013451                  TYPE    ,PRMPSP         ;TYPE "> " PROMPTER
                        ;READ AND CHECK INPUT, IF ANY
004737  032272                  JSR     PC,RDCHRS       ;READ INPUT LINE
014532                          DRVTST                  ;(↑C) RETURN ADDRESS FOR RDCHRS
016156                          ASKMDE                  ;(↑Z) RETURN ADDRESS FOR RDCHRS
017126                          SETPRM                  ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST 4 CHARACTERS TYPED
002005                          BGE     6$              ;BR IF AT LEAST 4 TYPED
104401  005262          4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
000761                          BR      SETPRM          ;BR TO ASK FOR INPUT AGAIN
020027  000013          6$:     CMP     R0,#11.         ;SEE IF ELEVEN OR LESS CHARS TYPED
003370                          BGT     4$              ;BR IF MORE THAN ELEVEN TYPED
```

```
                        ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
017172  005001          8$:     CLR     R1              ;INITIALIZE PARAMETER INDEX
017174  023761  005262  005776  8$:     CMP     BUFFO,PRMNEM(R1) ;TRY TO MATCH 2-CHAR MNEMONIC
017202  001411          BEQ     10$             ;BR IF PARAMETER FOUND IN TABLE
017204  062701  000002          ADD     #2,R1           ;INCREMENT INDEX
017210  010102          MOV     R1,R2           ;GET COPY OF INDEX
017212  062702  005646          ADD     #PRMLST,R2      ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
017216  022702  005674          CMP     #PROFLT,R2
017222  001364          BNE     8$              ;BR IF MORE TO CHECK YET
017224  000752          BR      4$              ;BR TO ECHO BAD INPUT
                        ;CHECK FOR VALID LINE FORMAT
017226  012702  000002  10$:    MOV     #2,R2           ;INITIALIZE BUFFER POINTER
017232  122762  000075  005262  CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS "="
017240  001411          BEQ     12$             ;BR IF IT IS "="
017242  122762  000040  005262  CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHAR IS A SPACE
017250  001340          BNE     4$              ;BR TO ECHO BAD INPUT
017252  005202          INC     R2              ;INCREMENT BUFFER POINTER
017254  122762  000075  005262  CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS "="
017262  001333          BNE     4$              ;BR TO ECHO INVALID INPUT
017264  005202          12$:    INC     R2              ;INCREMENT BUFFER POINTER
017266  020200          CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
017270  002330          BGE     4$              ;BR TO ECHO INVALID INPUT
017272  122762  000040  005262  CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHARACTER IS SPACE
017300  001003          BNE     14$             ;BR IF NOT A SPACE
017302  005202          INC     R2              ;INCREMENT BUFFER POINTER
017304  020200          CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
017306  002321          BGE     4$              ;BR IF NONE LEFT
017310  160200          14$:    SUB     R2,R0           ;SUBTRAC  POINTER FROM CHAR COUNT
017312  020027  000010          CMP     R0,#10          ;SEE HOW MANY CHARS LEFT
017316  003315          BGT     4$              ;BR IF TOO MANY LEFT
                        ;CHECK FOR LEGAL PARAMETER VALUE
017320  062702  005262          ADD     #BUFFO,R2       ;GET ADDRESS OF DIGITS
017324  010246          MOV     R2,-(SP)        ;PUT IT ON STACK FOR OCTBIN
017326  004737  053014          JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
017332  017152          4$              ;ERROR RETURN ADDR. FOR OCTBIN
017334  012637  005466          MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
017340  013737  053146  005470  MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
017346  004737  033276          JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAMETER VALUE
017352  017152          4$              ;ERROR RETURN ADDR. FOR CHKPRM
                        ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
017354  004737  032752          JSR     PC,MODP15
017360  000662          BR      SETPRM          ;RETURN TO ASK FOR ANOTHER PARAMETER



                        .SBTTL  TO - RUN (R) TESTS ROUTINE
                        ;*THIS ROUTINE IS ENTERED TO SET UP THE RUNNING OF TESTS,
                        ;*WHOSE PARAMETERS HAVE BEEN DETERMINED. IT FIRST ASKS
                        ;*FOR THE DESIRED NUMBER OF PROGRAM PASSES, AND STORES
                        ;*THIS IN $EOPCT FOR THE END OF PASS ROUTINE. THEN, THE
                        ;*DRIVE LIST IS SCANNED FOR THE FIRST DESIRED DRIVE, AND
                        ;*THE HEADER ON SECTOR 0 IS READ TO GET THE FORMAT FOR THIS
                        ;*DRIVE. THIS IS STORED IN THE FORMAT BYTE, AND A JUMP IS
                        ;*MADE TO THE FIRST TEST.
                        ;*THE END OF PASS ROUTINE RETURNS TO "NEWDRV" TO SELECT
                        ;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
```

```
                                        ;# (ALL DRIVES) IS COMPLETED.
                                        ;
017362                                  RUNTST:
                                        ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
017362  104401  011035                4$:     TYPE    ,ENTPAS         ;ASK FOR NUMBER OF PASSES
017366  004737  032272                        JSR     PC,RDCHRS       ;READ RESPONSE
017372  014532                          .       DRVTST                  ;(^C) RETURN ADDRESS
017374  016156                          .       ASKMOD                  ;(^Z) RETURN ADDRESS
017376  017362                          .       4$                      ;(^U) OR ERROR RETURN ADDRESS
017400  005700                          .       TST     R0              ;SEE IF NULL INPUT
017402  001403                          .       BEQ     6$              ;GO ASK AGAIN
017404  022700  000005                  .       CMP     #5,R0           ;SEE HOW MANY CHARS TYPED
017410  002005                          .       BGE     8$              ;BR IF 5 OR LESS
017412  104401  005262                6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
017416  104401  001314                        TYPE    ,$QUES
017422  000757                          .       BR      4$              ;GO ASK AGAIN
017424  012746  005262                8$:     MOV     #BUFFO,-(SP)    ;GET BUF ADDR ON STACK FOR OCTBIN
017430  004737  053014                        JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
017434  017412                          .       6$                      ;ERROR RETURN ADDRESS FOR OCTBIN
017436  012637  023540                        MOV     (SP)+,$EOPCT    ;SET DESIRED NO. OF PASSES (1-77777
017442  001763                          .       BEQ     6$              ;BR IF 0, TO ASK AGAIN
                                        ;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
017444  005037  001326                STPASS: CLR     $PASS           ;INIT. THE PASS NUMBER
017450  112737  000001  003110                MOVB    #1,ISTING       ;SET "RUNNING TESTS" FLAG
017456  012737  177777  005500        NEWPAS: MOV     #177777,DRIVE   ;INITIALIZE DRIVE NO. TO -1
017464  005037  001330                        CLR     $DEVCT          ;INIT APT DEVICE COUNT TO 0
                                        ;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
017470  012737  000000  177776        NEWDRV: MOV     #PRO,@#PS       ;RE-ESTABLISH PRIORITY 0
017476  012706  001100                        MOV     #STACK,SP       ;RESTORE THE STACK
017502  105037  003120                        CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
017506  005037  005522                        CLR     INTCHR          ;CLEAR TTY INPUT BUFFER WORD
017512  005237  005500                        INC     DRIVE           ;INCREMENT DRIVE NUMBER
017516  022737  000010  005500              CMP     #10,DRIVE       ;SEE IF DONE WITH THIS PASS
017524  001002                          .       BNE     2$              ;BR IF NOT DONE CHECKING LIST
017526  000137  023506                        JMP     DUNPAS          ;JUMP IF DONE WITH THIS PASS
017532  013700  005500                2$:     MOV     DRIVE,R0        ;GET NEW DRIVE NUMBER
017536  105760  005606                        TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
017542  001752                          .       BEQ     NEWDRV          ;BR IF NOT MARKED, TO CHECK ANOTHER
017544  105037  001102                        CLRB    $TSTNM          ;INIT TEST NUMBER TO 0
017550  005037  001304                        CLR     $TIMES          ;INIT ITERATION COUNT
017554  010037  001332                        MOV     R0,$UNIT        ;SET DRIVE NO. FOR APT
                                        ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
017560  004737  030776                        JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
017564  112765  000125  000001              MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
017572  004737  040774                        JSR     PC,DRVCAL       ;DO READ HEADER
017576  132765  000020  000007              BITB    #B.CFMT,P.CS1H(R5) ;COMPLEMENT THE FORMAT BIT
017604  001404                          .       BEQ     4$              ;  GIVEN TO THE CONTROLLER,
017606  142765  000020  000007              BICB    #B.CFMT,P.CS1H(R5) ;  AND DO READ HEADER. THIS
017614  000403                          .       BR      6$              ;  WILL CAUSE SECTOR 0 HEADER
017616  152765  000020  000007        4$:     BISB    #B.CFMT,P.CS1H(R5) ;  TO BE READ.
017624  004737  040774                6$:     JSR     PC,DRVCAL       ;READ HEADER 0
017630  105037  003115                        CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
017634  013737  005662  005506              MOV     S2,FS           ;INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
017642  013737  005664  005510              MOV     S3,LS
017650  005762  000024                        TST     RKDB(R2)        ;POP SILO ONE TIME
017654  032762  001000  000024              BIT     #BIT9,RKDB(R2)  ;TEST FORMAT BIT IN HEADER WORD 2
```

```
                                                        BEQ     8$                      ;BR IF 22 SECTOR FORMAT
                                                        BISB    #8.CFMT,FORMAT          ;SET 20 SECTOR FORMAT FOR THIS DRIVE
                                                        MOV     S0,FS                   ;INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
                                                        MOV     S1,LS
                                                8$:     MOV     ST,STALLS               ;SET NUMBER OF UNIT STALLS DESIRED
                                                        JSR     PC,INITSS               ;INIT THE S.S.
                                                        JSR     PC,REDBSF               ;READ BAD SECTOR FILE FOR THIS DRIVE
                                                        MOVB    DRIVE,DRVNO             ;GET DRIVE NO.
                                                        BISB    #'0,DRVNO               ;CONVERT TO ASCII
                                                        TYPE    ,TSTDRN                 ;TYPE "TESTING DRIVE X"
                                                        TST     $PASS                   ;SEE IF THIS IS FIRST PASS
                                                        BNE     10$                     ;BR IF NOT FIRST PASS
                                                        JSR     PC,DRVSER               ;TYPE "DRIVE SER. NO.  XXX"
                                                        JSR     PC,CRTSER               ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
                                                        BIT     #BIT4,CS                ;SEE IF BAD SECTORS SHOULD BE TYPED
                                                        BEQ     10$                     ;BR IF NOT
                                                        JSR     PC,TYPBSF               ;TYPE BAD SECTOR FILES
                                                10$:    CLR     INTCHR                  ;INIT. TTY INPUT CHAR BUFFER
```

```
                        ;;*****************************************************************
                        ;*TEST 1         OFFSET-TO-FAILURE MEASUREMENTS
                        ;*IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL
                        ;*WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC.
                        ;*THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH
                        ;* 167230(OCT) (TO ESTABLISH A KNOWN PATTERN)  BUT THEY ARE NOT TESTED.
                        ;*THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE
                        ;*OFFSETS, UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE
                        ;*OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE
                        ;*CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR
                        ;*MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS
                        ;*ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER : SECTORS FS AND LS ON
                        ;*CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR
                        ;*TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF
                        ;*INCREASING TRACKS, AS FOLLOWS :
                        ;*
                        ;* OFFSET-TO-FAILURE MEASUREMENTS :
                        ;*
                        ;* TRACK  CYLN  SECT  +OFST  -OFST
                        ;*                    (UIN)  (UIN)
                        ;*   X    XXX    XX   XXXX   XXXX
                        ;*   X    XXX    XX   XXXX   XXXX
                        ;*   X    XXX    XX   XXXX   XXXX
                        ;*          ETC.
                        ;*
                        ;*NOTE: IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS
                        ;*TO 10(DEC), FOR THIS TEST.
                        ;;*****************************************************************
000000 020002  000004           TST1:   SCOPE
000002 020004  012737  000001  001324          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
000010 020012  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
000014 020016  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
000020 020022  000137  021112                  JMP     TST2            ;JUMP TO NEXT TEST
                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
000024 020026  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
000030 020032  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                        ;INITIALIZE PARAMETERS, WRITE THE SECTORS, AND WRITE CHECK THEM
000034 020036  105737  003106                  TSTB    MDFLAG          ;SEE IF ADRS 200 DEFAULT RUN
000040 020042  001005                          BNE     3$              ;BR IF NOT
000042 020044  005037  005506                  CLR     FS              ;SET FS = 0
000046 020050  012737  000012  005510          MOV     #10.,LS         ;SET LS = 10(DEC)
000054 020056  013765  005646  000002  3$:     MOV     FC,P.CYLN(R5)   ;SET CYL = FC
000062 020064  113765  005506  000004          MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS
000070 020072  105065  000005                  CLRB    P.TRCK(R5)      ;SET TRACK = 0
000074 020076  013737  005650  005504          MOV     LC,CYLNDR
000102 020104  022737  000632  005650          CMP     #632,LC         ;SEE IF LC =632
000110 020112  001002                          BNE     4$              ;BR IF NOT 632
000112 020114  005337  005504                  DEC     CYLNDR          ;MAKE IT 631 (PRESERVE BSF)
000116 020120  012703  072307          4$:     MOV     #72307,R3       ;GET DATA PATTERN
000122 020124  004737  036050                  JSR     PC,WRTSEC       ;DO WRITE AND WRITE CHECK
000126 020130  012703  167230                  MOV     #167230,R3      ;DATA FOR AJACENT CYLS
000132 020134  005365  000002                  DEC     P.CYLN(R5)      ;NEXT LOWER CYL
000136 020140  100402                          BMI     6$              ;BR IF CYL NEGATIVE
000140 020142  004737  036050                  JSR     PC,WRTSEC       ;DO WRITE AND WRITE CHECK
000144 020146  062765  000002  000002  6$:     ADD     #2,P.CYLN(R5)   ;NEXT HIGHER CYL
000152 020154  022765  000632  000002          CMP     #632,P.CYLN(R5) ;SEE IF CYL = 632
```

```
                                                BNE     8$              ;BR IF NOT
                                                CMPB    #2,P.TRCK(R5)   ;SEE IF BSF
                                                BEQ     10$             ;BR IF YES, TO PROTECT IT
                                        8$:     JSR     PC,WRTSEC       ;DO WRITE AND WRITE CHECK
                                        10$:    DEC     P.CYLN(R5)      ;RESTORE CYL NO.
                                                INCB    P.TRCK(R5)      ;INCR TRACK
                                                CMPB    #3,P.TRCK(R5)   ;TRACK = 3 YET ?
                                                BNE     4$              ;BR IF NOT YET
                                                CMPB    P.SECT(R5),LS   ;SEE IF SECTOR = LS
                                                BEQ     14$             ;BR IF YES
                                                MOVB    LS,P.SECT(R5)   ;SET SECTOR = LS
                                        12$:    CLRB    P.TRCK(R5)      ;SET TRACK = 0
                                                BR      4$              ;BR TO CONTINUE WRITING
                                        14$:    CMP     P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
                                                BEQ     16$             ;BR IF YES
                                                MOV     CYLNDR,P.CYLN(R5) ;SET CYL = LC
                                                MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS AGAIN
                                                BR      12$             ;BR TO CONTINUE WRITING
                                        ;INITIALIZE PARAMETERS FOR OFFSET MEASUREMENTS
                                        16$:    BIT     #BIT1,CS        ;SEE IF REPORTS ARE INHIBITED
                                                BNE     18$             ;BR IF INHIBITED
                                                TYPE    ,OFSHED         ;TYPE HEADINGS
                                        18$:    MOVB    FT,P.TRCK(R5)   ;SET TRACK = FT
                                                MOV     FC,P.CYLN(R5)   ;SET CYLINDER = FC
                                                MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS
                                        ;LOAD THE R/W BUFFER FOR WRITE CHECKS
                                                MOV     #RWBUF,R0       ;SET BUFFER ADDRESS
                                                MOV     #400,R1         ;PREPARE TO LOAD 400(OCT) WORDS
                                        19$:    MOV     #072307,(R0)+   ;LOAD A WORD INTO BUFFER
                                                DEC     R1              ;DECR COUNTER
                                                BNE     19$             ;BR IF NOT DONE YET
                                        20$:    CLR     R4              ;INIT (+) OFFSET VALUE
                                                CLR     R0              ;INIT OFFSET NUMBER TO 0
                                                MOV     #-1,PLOFST      ;INITIALIZE CURRENT FAILING (+) OFFSET
                                                MOV     #-1,NGOFST      ;INITIALIZE CURRENT FAILING (-) OFFSET
                                        ;PERFORM OFFSETS AND WRITE CHECKS
                                        22$:    MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
                                                JSR     PC,DRVCAL       ;PERFORM SEEK
                                                MOVB    R0,P.OFST(R5)   ;SET CURRENT OFFSET
                                                MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
                                                JSR     PC,DRVCAL       ;PERFORM OFFSET
                                                MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
                                                MOV     #WRCKHD,A.ABNL  ;SET SPECIAL ERROR HANDLER ADDRESS
                                                CLRB    WCEFLG          ;CLEAR WRITE CHECK ERROR FLAG
                                                JSR     PC,DRVCAL       ;PERFORM A WRITE CHECK
                                                MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADDRESS
                                                MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR CMND
                                                JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
                                                TSTB    WCEFLG          ;SEE IF A WRITE CHECK ERROR OCCURRED
                                                BNE     26$             ;BR IF AN ERROR OCCURRED
                                                MOV     R0,R1           ;GET A COPY OF CURRENT OFFSET
                                                INC     R0              ;INCREMENT OFFSET
                                                ADD     #25.,R4         ;INCR OFFSET VALUE BY 25(DEC) UIN
                                                BIC     #177700,R1      ;MASK FOR MAGNITUDE BITS
                                                CMP     #60,R1          ;SEE IF MAX OFFSET APPLIED IN THIS DIRECTION
                                                BNE     22$             ;BR IF NOT YET
```

```
 5598  020514  032700  000200                BIT     #BIT7,R0        ;SEE IF NEG OFFSETS DONE YET
 5599  020520  001015                        BNE     32$             ;BR IF YES, TO PRINT CURRENT SECTOR RESULT
 5600  020522  012700  000200          24$:  MOV     #200,R0         ;INIT FOR NEG OFFSETS
 5601  020526  005004                        CLR     R4              ;INIT (-) OFFSET VALUE TO 0
 5602  020530  000717                        BR      22$             ;BR TO APPLY NEG OFFSETS
 5603  020532  032700  000200          26$:  BIT     #BIT7,R0        ;SEE IF NEG OFFSETS DONE YET
 5604  020536  001403                        BEQ     28$             ;BR IF NOT YET
 5605  020540  010437  005546                MOV     R4,NGOFST       ;SAVE THIS FAILING (-) OFFSET VALUE
 5606  020544  000403                        BR      32$             ;BR TO PRINT CURRENT SECTOR RESULT
 5607  020546  010437  005544          28$:  MOV     R4,PLOFST       ;SAVE THIS FAILING (+) OFFSET VALUE
 5608  020552  000763                        BR      24$             ;BR TO APPLY NEGATIVE OFFSETS
 5609  020554  032737  000002  005670  32$:  BIT     #BIT1,CS        ;SEE IF REPORTS INHIBITED
 5610  020562  001051                        BNE     43$             ;BR IF INHIBITED
 5611                          ;TYPE OFFSET RESULTS FOR THIS SECTOR
 5612  020564  116501  000005                MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
 5613  020570  010146                        MOV     R1,-(SP)        ;PUT IT ON STACK
 5614  020572  104403                        TYPOS                   ;TYPE IT
 5615  020574     003                        .BYTE   3               ;3 DIGITS
 5616  020575     000                        .BYTE   0               ;SUPPRESS
 5617  020576  104401  013424                TYPE    ,SPACE3
 5618  020602  016546  000002                MOV     P.CYLN(R5),-(SP) ;PUT CYL ON STACK
 5619  020606  104403                        TYPOS                   ;TYPE IT
 5620  020610     004                        .BYTE   4               ;4 DIGITS
 5621  020611     000                        .BYTE   0               ;SUPPRESS
 5622  020612  104401  013425                TYPE    ,SPACE2
 5623  020616  116501  000004                MOVB    P.SECT(R5),R1   ;GET SECTOR
 5624  020622  010146                        MOV     R1,-(SP)        ;PUT IT ON STACK
 5625  020624  104404                        TYPON                   ;TYPE SECTOR
 5626  020626  104401  013426                TYPE    ,SPACE1
 5627  020632  005737  005544                TST     PLOFST          ;SEE IF THERE WAS A FAILING (+) OFFSET
 5628  020636  100003                        BPL     36$             ;BR IF THERE WAS
 5629  020640  104401  011271                TYPE    ,NOFALS         ;TYPE "  NONE"
 5630  020644  000403                        BR      38$             ;CONTINUE
 5631  020646  013746  005544          36$:  MOV     PLOFST,-(SP)    ;PUT (+) OFFSET VALUE ON STACK
 5632  020652  104405                        TYPDS                   ;TYPE IT IN DECIMAL
 5633  020654  104401  013426          38$:  TYPE    ,SPACE1
 5634  020660  005737  005546                TST     NGOFST          ;SEE IF THERE WAS A FAILING (-) OFFSET
 5635  020664  100003                        BPL     40$             ;BR IF THERE WAS
 5636  020666  104401  011271                TYPE    ,NOFALS         ;TYPE "  NONE"
 5637  020672  000403                        BR      42$             ;CONTINUE
 5638  020674  013746  005546          40$:  MOV     NGOFST,-(SP)    ;PUT (-) OFFSET VALUE ON STACK
 5639  020700  104405                        TYPDS                   ;TYPE IT IN DECIMAL
 5640  020702  104401  001315          42$:  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 5641                          ;SET UP TO TEST NEXT SECTOR
 5642  020706  126537  000004  005510  43$:  CMPB    P.SECT(R5),LS   ;SEE IF SECTOR = LS
 5643  020714  001404                        BEQ     44$             ;BR IF YES
 5644  020716  113765  005510  000004        MOVB    LS,P.SECT(R5)   ;SET SECTOR = LS
 5645  020724  000611                        BR      20$             ;GO TEST THIS SECTOR
 5646  020726  026537  000002  005504  44$:  CMP     P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
 5647  020734  001410                        BEQ     48$             ;BR IF YES
 5648  020736  013765  005504  000002        MOV     CYLNDR,P.CYLN(R5) ;SET CYL = LC
 5649  020744  113765  005506  000000  46$:  MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS
 5650  020752  000137  020350                JMP     20$             ;GO TEST THIS SECTOR
 5651  020756  105265  000005          48$:  INCB    P.TRCK(R5)      ;INCREMENT TRACK
 5652  020762  122765  000003  000005        CMPB    #3,P.TRCK(R5)   ;SEE IF TRACK = 3
 5653  020770  001404                        BEQ     50$             ;BR IF YES
```

```
005654  020772  013765  005646  000002          MOV     FC,P.CYLN(R5)   ;SET CYL = FC
005655  021000  000761                          BR      46$             ;GO TEST THIS SECTOR
005656  021002  112765  000113  000001  50$:    MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
005657  021010  004737  040774                  JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
005658  021014  013737  005662  005506          MOV     52,FS           ;RESTORE FS,LS FOR 22 SECTORS
005659  021022  013737  005664  005510          MOV     53,LS
005660  021030  105737  003115                  TSTB    FORMAT          ;SEE IF 22 SECTORS
005661  021034  001406                          BEQ     54$             ;BR IF YES
005662  021036  013737  005656  005506          MOV     50,FS           ;RESTORE FS,LS FOR 20 SECTORS
005663  021044  013737  005660  005510          MOV     51,LS
005664  021052                          54$:
005665  021052  000137  021112                  JMP     TST2            ;JUMP TO NEXT TEST
005666
005667                                  ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER, USED WHEN A
005668                                  ;*   "DATA TYPE" ERROR IS EXPECTED (AND VALID).
005669  021056  032765  040000  000020  WRCKHD: BIT     #WCE,P.CS2(R5)  ;SEE IF WRITE CHECK ERROR OCCURRED
005670  021064  001006                          BNE     4$              ;BR IF WCE
005671                                  ;CHECK FOR OTHER "DATA TYPE" ERROR
005672  021066  032765  130400  000034          BIT     #HVRC!DTE!OPI!DCK,P.ER(R5)
005673  021074  001002                          BNE     4$              ;BR IF A DATA ERROR OCCURRED
005674  021076  000137  042466                  JMP     ERRHDL          ;GO HANDLE OTHER ERROR
005675  021102  105237  003131          4$:     INCB    WCEFLG          ;SET WRITE CHECK ERROR FLAG
005676  021106  000137  044702                  JMP     RETNML          ;TAKE NORMAL RETURN
005677
005678
005679
005680                                  ;;*****************************************************************
005681                                  ;*TEST 2        NPR/MEMORY WORD ADDRESSING TEST
005682                                  ;*IN THIS TEST, MEMORY WORD ADDRESSING CAPABILITY DURING RK06 NPR'S,
005683                                  ;*IS TESTED.
005684                                  ;*BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
005685                                  ;*THEN, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK
005686                                  ;*BEYOND ADDRESS RWBUF+6000(OCTAL) , WRITE UNIQUE NUMBERS
005687                                  ;*(STARTING WITH 1), INTO EACH OF UP TO 64K WORDS (DEPENDING UPON THE
005688                                  ;*AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES
005689                                  ;*WITHIN THE 64K BLOCK. NEXT, WRITE THE 64K WORDS (MAX) ONTO THE DISK
005690                                  ;*AT ADDRESS FC,FT,FS (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW
005691                                  ;*OR DESTRUCTION OF THE BAD SECTOR FILE). THEN ZERO THE ENTIRE BLOCK
005692                                  ;*IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL
005693                                  ;*ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING
005694                                  ;*VIRTUAL (IF MEM. MGT.) AS WELL AS PHYSICAL ADDRESSES, AND THE GOOD
005695                                  ;*AND BAD DATA, FOR UP TO THE FIRST 10(DEC) FAILING LOCATIONS.
005696                                  ;* IF MEMORY MANAGEMENT IS INSTALLED AND THERE IS ADDITIONAL MEMORY TO
005697                                  ;*EXERCISE, REPEAT THE ABOVE WORD ADDRESSING TEST, FOR EACH OF THE
005698                                  ;*REMAINING 64K MEMORY BLOCKS.
005699                                  ;;*****************************************************************
005700  021112  000004                  TST2:   SCOPE
005701  021114  012737  000002  001324          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
005702  021122  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
005703  021126  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
005704  021132  000137  021422                  JMP     TST3            ;JUMP TO NEXT TEST
005705                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
005706  021136  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
005707  021142  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
005708                                  ;SAVE XXDP LOADER, IF PRESENT
005709  021146  004737  030456                  JSR     PC,FNDXDP       ;COMPUTE STARTING ADR OF XXDP
```

```
5710  021152  012737  064640  005540          MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADR
5711  021160  005037  005542                  CLR     XDPSAV+2
5712  021164  004737  030530                  JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
5713  021170  112737  000001  003124          MOVB    #1,XOVLAD       ;SET INDICATORS
5714  021176  112737  000001  003125          MOVB    #1,XDPSVD
5715  021204  012737  177777  005534          MOV     #-1,PATRN       ;SET FLAG FOR SVPRMS
5716                                  ;INIT PARAMETERS, BEGIN TESTING
5717  021212  004737  037100                  JSR     PC,SETUPM       ;INIT PARAMS
5718  021216  004737  033526          4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WORD COUNT FOR THIS MA
5719  021222  005701                          TST     R1              ;SEE IF ALL MEMORY TESTED YET
5720  021224  100467                          BMI     50$             ;BR IF ALL DONE
5721  021226  005065  000012                  CLR     P.WC(R5)        ;INIT WORD COUNT TO 65,536(DEC)
5722  021232  005701                          TST     R1              ;SEE IF WRD CNT SHOULD BE 65,536
5723  021234  001003                          BNE     6$              ;BR IF YES
5724  021236  005400                          NEG     R0
5725  021240  010065  000012                  MOV     R0,P.WC(R5)     ;SET WORD COUNT
5726  021244  013765  005576  000010  6$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
5727  021252  013701  005600                  MOV     MA+2,R1         ;GET MA BITS 16-21
5728  021256  042701  177774                  BIC     #177774,R1      ;MASK FOR LO 2 BITS
5729  021262  150165  000007                  BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
5730  021266  005737  056246                  TST     $KT11           ;SEE IF MEM MGT PRESENT
5731  021272  100010                          BPL     14$             ;BR IF NOT
5732                                  ;SET UP MEM MGT
5733  021274  013737  005576  005602          MOV     MA,PMA          ;SET BUFFER ADDRESS
5734  021302  013737  005600  005604          MOV     MA+2,PMA+2
5735  021310  004737  030334                  JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBLS MAP
5736                                  ;PERFORM THE TESTS
5737  021314  012737  021314  001110  14$:     MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
5738  021322  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5739  021326  004737  040304                  JSR     PC,SVPRMS       ;SAVE PARAMS FOR THIS XFER
5740  021332  004737  036144                  JSR     PC,LDMEM1       ;LOAD THIS MEM BLK WITH INCREMENTING DATA
5741  021336  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5742  021344  004737  040416                  JSR     PC,TRNSFR       ;WRITE THE DATA
5743  021350  005003                          CLR     R3              ;SET DATA WORD = 0
5744  021352  004737  036156                  JSR     PC,LDMEM2       ;LOAD MEM BLK WITH ALL ZEROS
5745  021356  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
5746  021364  004737  040416                  JSR     PC,TRNSFR       ;READ THE DATA FROM DISK
5747  021370  004737  036310                  JSR     PC,CKMEM1       ;PERFORM SOFTWARE COMPARE OF DATA
5748  021374  104411                          SCOPER                  ;CHECK FOR INTERNAL LOOP ON ERROR
5749                                  ;GET NEW MA FOR NEXT TRANSFER
5750  021376  004737  037306                  JSR     PC,INCRMA       ;COMPUTE NEXT MA
5751  021402  000705                          BR      4$              ;GO SEE IF MORE MEMORY TO TEST
5752  021404  004737  030536          50$:    JSR     PC,GETXDP       ;RESTORE XXDP LOADER
5753  021410  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
5754  021414  001402                          BEQ     54$             ;BR IF NOT
5755  021416  005037  172516                  CLR     @#SR3           ;DISABLE UNIBUS MAP
5756  021422                          54$:
5757
5758
5759
5760                                  ;;*****************************************************************
5761                                  ;*TEST 3        NPR/MEMORY BLOCK ADDRESSING TEST
5762                                  ;*IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06
5763                                  ;*NPR'S IS TESTED.
5764                                  ;*BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
5765                                  ;*MEMORY WILL BE TESTED STARTING AT THE FIRST ADRS OF THE NEXT 1K
```

```
                                        ;*BLOCK BEYOND RWBUF+6000(OCTAL).
                                        ;*THE TEST BEGINS BY WRITING A SINGLE NUMBER INTO ALL LOCATIONS OF
                                        ;*EACH MEMORY BLOCK OF UP TO 64K WORDS. THIS NUMBER UNIQUELY DEFINES
                                        ;*EACH BLOCK. THEN WRITE EACH BLOCK ONTO DISK AT ADDRESS FC,FS,FT
                                        ;*(SCALED, IF NECESSARY TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
                                        ;*SECTOR FILE). ZERO THE BLOCK IN MEMORY, AND THEN READ THE DATA
                                        ;*BACK INTO THE PROPER PHYSICAL ADDRESSES. DO THIS FOR ALL 64K BLOCKS,
                                        ;*AND THEN COMPARE THE CONTENTS OF ALL THE MEMORY TO EXPECTED VALUES.
                                        ;*TYPE FAILURES AS DESCRIBED ABOVE FOR THE FIRST 10(DEC) FAILING
                                        ;*LOCATIONS IN EACH 64K MEMORY BLOCK.
                                        ;;*****************************************************************
021422  000004                  TST3:   SCOPE
021424  012737  000003  001324          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
021432  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021436  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
021442  000137  021762                  JMP     TST4            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
021446  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
021452  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;SAVE XXDP LOADER, IF PRESENT
021456  004737  030456                  JSR     PC,FNDXDP       ;COMPUTE STARTING ADRS OF XXDP
021462  012737  064640  005540          MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADDRESS
021470  005037  005542                  CLR     XDPSAV+2
021474  004737  030530                  JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
021500  112737  000001  003124          MOVB    #1,XOVLAD       ;SET INDICATORS
021506  112737  000001  003125          MOVB    #1,XDPSVD
021514  012737  177777  005534          MOV     #-1,PATRN       ;SET FLAG FOR SVPRMS
                                        ;INIT PARAMETERS, WRITE AND READ BACK ALL MEMORY BLOCKS
021522  005037  005524                  CLR     SELECT          ;INIT COMPARE FLAG
021526  004737  037100          2$:     JSR     PC,SETUPM       ;INIT PARAMS
021532  012704  000001                  MOV     #1,R4           ;INIT DATA WORD
021536  004737  033526          4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WRD CNT FOR THIS MA
021542  005701                          TST     R1              ;SEE IF ALL MEMORY TESTED YET
021544  100473                          BMI     30$             ;BR IF DONE WRITING AND READING
021546  005065  000012                  CLR     P.WC(R5)        ;INIT WRD CNT TO 65,536(DEC)
021552  005701                          TST     R1              ;SEE IF WRD CNT SHOULD BE 65536
021554  001003                          BNE     6$              ;BR IF YES
021556  005400                          NEG     R0
021560  010065  000012                  MOV     R0,P.WC(R5)     ;SET WORD COUNT
021564  013765  005576  000010  6$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
021572  013701  005600                  MOV     MA+2,R1         ;GET MA BITS 16-21
021576  042701  177774                  BIC     #177774,R1      ;MASK FOR LO 2 BITS
021602  150165  000007                  BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
021606  005737  056246                  TST     $KT11           ;SEE IF MEM MGT PRESENT
021612  100010                          BPL     14$             ;BR IF NO
                                        ;SET UP MEMORY MANAGEMENT
021614  013737  005576  005602          MOV     MA,PMA          ;SET BUFFER ADDRESS
021622  013737  005600  005604          MOV     MA+2,PMA+2
021630  004737  030334                  JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBUS MAP
021634  005737  005524          14$:    TST     SELECT          ;SEE IF COMPARES SHOULD BE DONE YET
021640  001026                          BNE     20$             ;BR IF YES
                                        ;WRITE AND READ THIS MEM BLK ON RK06
021642  004737  040304          16$:    JSR     PC,SVPRMS       ;SAVE PARAMS FOR THIS XFER
021646  010403                          MOV     R4,R3           ;GET DATA WORD
021650  004737  036156                  JSR     PC,LDMEM2       ;LOAD MEMORY BLK WITH THIS WORD
021654  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
```

```
                                        JSR     PC,TRNSFR       ;WRITE THE DATA
                                        CLR     R3
                                        JSR     PC,LDMEM2       ;ZERO THE BLK IN MEMORY
                                        MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
                                        JSR     PC,TRNSFR       ;READ THIS DATA BACK
                                        INC     R4              ;INCREMENT THE DATA WORD
                                        JSR     PC,INCRMA       ;COMPUTE NEW MA TO TRY
                                        BR      4$              ;GO WRITE AND READ NEXT BLK
                                ;PERFORM SOFTWARE COMPARES OF ALL MEMORY BLOCKS
                                20$:    MOV     R4,R3           ;GET DATA WORD
                                        JSR     PC,CKMEM2       ;COMPARE THIS BLK TO GOOD DATA WORD
                                        INC     R4              ;INCREMENT THE DATA WORD
                                        JSR     PC,INCRMA       ;GET NEW MA TO TRY
                                        BR      4$              ;GO COMPARE NEXT MEM BLK

                                30$:    COM     SELECT          ;COMPLEMENT THE FLAG
                                        BEQ     50$             ;BR IF ALL DONE NOW
                                        BR      2$              ;BR IF COMPARES SHOULD BE DONE NOW
                                50$:    JSR     PC,GETXDP       ;RESTORE SAVED XXDP
                                        TSTB    JBMPRS          ;SEE IF UNIBUS MAP PRESENT
                                        BEQ     54$             ;BR IF NOT
                                        CLR     @#SR3           ;DISABLE UNIBUS MAP
                                54$:

                                ;;*****************************************************************
                                ;*TEST 4        NPR/MEMORY DATA PATTERN TEST
                                ;*BEFORE TESTING, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS
                                ;*RWBUF. THEN, ALL PHYSICAL MEMORY LOCATIONS , STARTING AT THE
                                ;*FIRST ADRS OF THE NEXT 1K MEMORY BLOCK BEYOND RWBUF+6000(OCT),
                                ;* ARE EXERCISED WITH UP TO 16 DATA PATTERNS
                                ;* CHOSEN IN PARAMETER PT. (THESE ARE THE SAME PATTERNS
                                ;*PROVIDED FOR USE IN THE READ/WRITE DATA TEST). IF PT = 0, THE
                                ;*DATA PATTERNS WILL DEFAULT TO PATS 08,09,10,AND 11.
                                ;*MEMORY IS EXERCISED IN BLOCKS OF UP TO 64K WORDS, STARTING AT
                                ;*THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE BUFFER.
                                ;*EACH BLOCK IS LOADED WITH DATA, AND WRITTEN ONTO DISK AT ADDRESS
                                ;*FC,FT,FS (SCALED TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
                                ;*SECTOR FILE). THEN THE MEMORY BLOCK IS ZEROED AND READ BACK AND
                                ;*COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN,
                                ;*POSSIBLY INCLUDING THE USER-DEFINED PATTERN 15, IF SPECIFIED.
                                ;;*****************************************************************
021762  000004                  TST4:   SCOPE
021764  012737  000004  001324          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
021772  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
022000  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022004  000137  022332                  JMP     TST5            ;JUMP TO NEXT TEST
                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022010  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022014  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                ;SAVE XXDP LOADER, IF PRESENT
022020  004737  030456                  JSR     PC,FNDXDP       ;COMPUTE STARTING ADR OF XXDP
022024  012737  064640  005540          MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADR
022032  005037  005542                  CLR     XDPSAV+2
022036  004737  030530                  JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
```

```
5878  022040  112737  000001  003124          MOVB    #1,XOVLAD       ;SET INDICATORS
5879  022046  112737  000001  003125          MOVB    #1,XDPSVD
5880  022054  004737  034270                  JSR     PC,LODP14       ;GENERATE PSEUDO-RAND PAT 14
5881  022060  013737  005666  005534          MOV     PT,PATRN        ;GET A COPY OF  PT
5882  022066  001003                          BNE     2$              ;BR IF PT NON-ZERO
5883  022070  012737  007400  005534          MOV     #7400,PATRN     ;SET DEFLT PATRNS = PAT 8,9,10,11
5884                        ;INIT PARAMETERS, BEGIN TESTING
5885  022076  004737  037100          2$:     JSR     PC,SETUPM       ;INIT PARAMS
5886  022102  004737  033526          4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WORD COUNT FOR THIS MA
5887  022106  005701                          TST     R1              ;SEE IF ALL MEMORY TESTED YET
5888  022110  100501                          BMI     50$             ;BR IF ALL DONE
5889  022112  005065  000012                  CLR     P.WC(R5)        ;INIT WORD COUNT TO 65,536(DEC)
5890  022116  005701                          TST     R1              ;SEE IF WRD CNT SHOULD BE 65,536
5891  022120  001003                          BNE     6$              ;BR IF YES
5892  022122  005400                          NEG     R0
5893  022124  010065  000012                  MOV     R0,P.WC(R5)     ;SET WORD COUNT
5894  022130  013765  005576  000010  6$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
5895  022136  013701  005600                  MOV     MA+2,R1         ;GET MA BITS 16-21
5896  022142  042701  177774                  BIC     #177774,R1      ;MASK FOR LO 2 BITS
5897  022146  150165  000007                  BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
5898  022152  005737  056246                  TST     $KT11           ;SEE IF MEM MGT PRESENT
5899  022156  100010                          BPL     14$             ;BR IF NOT
5900                        ;SET UP MEM MGT
5901  022160  013737  005576  005602          MOV     MA,PMA          ;SET BUFFER ADDRESS
5902  022166  013737  005600  005604          MOV     MA+2,PMA+2
5903  022174  004737  030334                  JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5904  022200  012704  000001          14$:    MOV     #1,R4           ;SET PATTERN BIT POINTER
5905  022204  030437  005534          16$:    BIT     R4,PATRN        ;SEE IF THIS PATTERN IS CHOSEN
5906  022210  001003                          BNE     20$             ;BR IF YES
5907  022212  006304                  18$:    ASL     R4              ;SHIFT TO  POINT TO NEXT PATTERN
5908  022214  001434                          BEQ     24$             ;BR IF NO MORE LEFT TO CHECK
5909  022216  000772                          BR      16$             ;GO CHECK FOR ANOTHER PATTERN
5910  022220  010401                  20$:    MOV     R4,R1           ;GET A COPY OF BIT POINTER
5911                        ;PERFORM THE TESTS
5912  022222  012737  022222  001110  22$:    MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
5913  022230  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5914  022234  004737  040304                  JSR     PC,SVPRMS       ;SAVE THE PARAMS FOR THIS XFER
5915  022240  004737  034504                  JSR     PC,LODBUF       ;LOAD THIS MEM BLK WITH SELECTED DATA
5916  022244  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5917  022252  004737  040416                  JSR     PC,TRNSFR       ;WRITE THE DATA
5918  022256  005003                          CLR     R3              ;SET DATA WORD = 0
5919  022260  004737  036156                  JSR     PC,LDMEM2       ;LOAD MEM BLK WITH ALL ZEROS
5920  022264  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
5921  022272  004737  040416                  JSR     PC,TRNSFR       ;READ THE DATA FROM DISK
5922  022276  004737  034732                  JSR     PC,CMPBUF       ;PERFORM SOFTWARE COMPARE OF DATA
5923  022302  104411                          SCOPER                  ;CHECK FOR INTERNAL LOOP ON ERROR
5924  022304  000742                          BR      18$             ;CHECK FOR  NEXT PATTERN
5925                        ;GET NEW MA FOR NEXT TRANSFER
5926  022306  004737  037306          24$:    JSR     PC,INCRMA       ;COMPUTE NEXT MA
5927  022312  000673                          BR      4$              ;GO SEE IF MORE MEMORY TO TEST
5928  022314  004737  030536          50$:    JSR     PC,GETXDP       ;RESTORE XXDP LOADER
5929  022320  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
5930  022324  001402                          BEQ     54$             ;BR IF NOT PRESENT
5931  022326  005037  172516                  CLR     @#SR3           ;DISABLE UNIBUS MAP
5932  022332                          54$:
5933
```

```
                                ;;*****************************************************************
                                ;*TEST 5        UNIBUS CONTENTION TEST
                                ;*THE PURPOSE OF THIS TEST IS TO EXERCISE THE NPR CONTROL LOGIC
                                ;*WITHIN THE CONTROLLER IN SUCH A MANNER THAT A CONTENTION FOR
                                ;*UNIBUS CYCLES ARISES BETWEEN THE PROCESSOR AND THE RK06.
                                ;*
                                ;*FIRST, A WRITE DATA IS BEGUN AT ADDRESS FC,FT,SECTOR 0, WITH
                                ;*THE BUS ADDRESS INCREMENT INHIBIT BIT (BAI) SET IN THE CONTROLLER.
                                ;*USING A WORD COUNT OF 12000(OCT) OR 13000(OCT) (DEPENDING ON THE
                                ;*FORMAT), THE ENTIRE TRACK IS WRITTEN WITH THE FIRST WORD OF PATTERN
                                ;*15 (IF SELECTED) OR THE FIRST WORD OF PAT 13 (BY DEFAULT).
                                ;* THEN, AN INSTRUCTION LOOP IS ENTERED IN WHICH A NON-
                                ;*EXISTENT UNIBUS DEVICE ADDRESS IS REFERENCED ONCE PER PASS. THIS
                                ;*WILL RESULT IN A PROCESSOR SIEZURE OF THE UNIBUS, FOR 5-20 USEC.
                                ;*(DEPENDING ON THE PROCESSOR), FOLLOWED BY A NON-EXISTENT MEMORY
                                ;*PROCESSOR TRAP, ONCE PER PASS. THUS, FOR EACH PASS THROUGH THE
                                ;*INSTRUCTION LOOP THE CONTROLLER IS FORCED TO LOSE FROM ONE
                                ;*TO FOUR NPR BUS CYCLES. THE EXECUTION TIME OF THE LOOP DETERMINES THE
                                ;*REPETITION RATE OF THE NON-EXISTENT MEMORY REFERENCES, AND THIS
                                ;*IS CHOSEN TO ALLOW ENOUGH NPR'S SO THAT THERE SHOULD BE NO
                                ;*DATA LATE ERRORS IN A FAULT-FREE CONTROLLER.
                                ;*THE DATA IS THEN WRITE CHECKED FOR VERIFICATION, AND THE ABOVE
                                ;*TEST IS REPEATED USING A READ DATA COMMAND. NO ERRORS ARE EXPECTED.
                                ;*
                                ;*THE ABOVE OPERATIONS ARE REPEATED 15 MORE TIMES, USING THE NEXT WORD
                                ;*OF THE PROPER PATTERN (13 OR 15) EACH TIME, TO WRITE AND READ
                                ;*THE ENTIRE TRACK. IF A DATA LATE ERROR OCCURS ON A GIVEN TRANSFER,
                                ;*THAT DATA IS NOT WRITE-CHECKED OR COMPARED.
                                ;;*****************************************************************
022332  000004                  TST5:   SCOPE
022334  012737  000005  00:324         MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
022342  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
022346  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022352  000137  022636                  JMP     TST6            ;JUMP TO NEXT TEST
                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022356  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022362  004737  027202                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                ;INITIALIZE PARAMETERS
022366  013765  005646  000002          MOV     FC,P.CYLN(R5)   ;SET CYL = FC
022374  113765  005652  000005          MOVB    FT,P.TRCK(R5)   ;SET TRACK = FT
022402  105065  000004                  CLRB    P.SECT(R5)      ;SET SECTOR = 0
022406  012765  165000  000012          MOV     #-13000,P.WC(R5) ;INIT WORD COUNT FOR FULL TRACK
022414  105737  003115                  TSTB    FORMAT          ;DETERMINE THE FORMAT
022420  001403                          BEQ     4$              ;BR IF 22 SECTORS
022422  012765  166000  000012          MOV     #-12000,P.WC(R5) ;SET WORD COUNT FOR FULL TRACK
022430  052765  100000  000014  4$:     BIS     #DTBAII,P.PRST(R5) ;SET BUS ADDRESS INCREMENT INHIBIT
                                ;WRITE DATA WITH ALLOWABLE UNIBUS CONTENTION, AND WRITE CHECK IT
022436  012700  006764                  MOV     #PAT15,R0       ;SET DATA PATTERN ADDRESS
022442  005737  005666                  TST     PT              ;SEE IF USER-DEFINED PATTERN 15 SELECTED
022446  100402                          BMI     14$             ;BR IF YES
022450  012700  006664                  MOV     #PAT13,R0       ;SET DATA PATTERN 13 ADDRESS
022454  012701  000020          14$:    MOV     #16.,R1         ;SET COUNTER = 16
022460  012703  000024                  MOV     #20.,R3         ;SET NON-EXISTENT MEMORY TIMER
```

```
                                16$:    MOV     R0,P.BAL0(R5)   ;SET BA BITS
                                        CLRB    DLTFLG          ;CLEAR DATA LATE FLAG
                                        MOVB    #1,NORTRY       ;SET "NO-RETRY" FLAG
                                        MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
                                        JSR     PC,REFNEM       ;WRITE DATA DURING NEM REF.
                                        TSTB    DLTFLG          ;SEE IF DATA LATE ERROR OCCURRED
                                        BNE     18$             ;BR IF NOT
                                        BIT     #BSERR,RECODE   ;SEE IF BAD SECTOR ERROR
                                        BNE     24$             ;BR IF YES
                                        MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
                                        JSR     PC,DRVCAL       ;DO WRITE CHECK
                                ;READ DATA WITH ALLOWABLE UNIBUS CONTENTION, AND COMPARE LAST WORD
                                18$:    MOV     #RWBUF,P.BAL0(R5) ;SET BA = RWBUF ADDRESS
                                        CLR     RWBUF           ;CLEAR THE BUFFER WORD
                                        MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
                                        JSR     PC,REFNEM       ;READ DATA DURING NEM REF.
                                        TSTB    DLTFLG          ;SEE IF DATA LATE ERROR OCCURRED
                                        BNE     20$             ;BR IF YES
                                        CMP     (R0)+,RWBUF     ;COMPARE LAST DATA WORD
                                        BEQ     20$             ;BR IF EQUAL
                                        JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
                                        MOV     -2(R0),$REG5    ;GOOD DATA WORD
                                        MOV     RWBUF,$REG6     ;BAD DATA WORD
                                        ERROR   110             ;"READ ERROR WHILE BAI SET"
                                ;SET UP FOR NEXT PASS THROUGH LOOP
                                20$:    DEC     R1              ;SEE IF ALL DONE YET
                                        BEQ     24$             ;BR IF YES
                                        JMP     16$             ;JUMP TO CONTINUE TESTING
                                24$:


                                ;;*****************************************************************
                                ;*TEST 6         MULTI-DRIVE INTERFERENCE TEST
                                ;*THIS TEST PERFORMS MULTI-DRIVE POSITIONING OPERATIONS, WHILE A
                                ;*LARGE DATA TRANSFER IS IN PROGRESS ON THE DRIVE UNDER TEST, FOR
                                ;*THE PURPOSE OF DETECTING PROBLEMS OF CONCURRENT DRIVE OPERATION.
                                ;*THE TEST IS RUN ONLY IF THERE IS MORE THAN 1 DRIVE ON THE SUBSYSTEM.
                                ;*
                                ;*THE TEST PROCEEDS AS FOLLOWS :  IT IS FIRST DETERMINED WHICH DRIVE(S)
                                ;*(BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES
                                ;*ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH.NEXT, A SEEK IS
                                ;*DONE TO CYLINDER FC (SCALED, IF NECESSARY,) ON THE DRIVE UNDER
                                ;*TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH
                                ;*TO A DIFFERENT PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE
                                ;*FROM 10-67 MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN,
                                ;*A WRITE DATA COMMAND IS BEGUN ON THE DRIVE UNDER TEST AT THE CURRENT
                                ;*CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT
                                ;*INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616
                                ;*(DEC) WORDS IF 20 SECTOR FORMAT OR 17,152(DEC) WORDS IF 22 SECTOR
                                ;*FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA.
                                ;*THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN
                                ;*THAT TIME, ALL THE OTHER DRIVES SHOULD HAVE COMPLETED POSITIONING.
                                ;*NEXT, A WRITE CHECK IS DONE TO VERIFY THE DATA WRITTEN ON THE DRIVE
                                ;*UNDER TEST, AND A CHECK IS MADE TO INSURE THAT ALL OTHER DRIVES WHICH
                                ;*PERFORMED SEEKS WERE ABLE TO COMPLETE THEM SUCCESSFULLY. BY CHECKING
```

```
                                        ;*ATTENTION BITS AND COMPARING DRIVE CYLINDER ADDRESSES TO EXPECTED
                                        ;*VALUES.
                                        ;;*****************************************************************
022636  000004                  TSTE:   SCOPE
022640  012737  000006  001324          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
022646  004737  032130                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
022652  004737  032176                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022656  000137  023474                  JMP     $EOP            ;JUMP TO END-OF-PASS ROUTINE
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022662  004737  030776                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022666  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;CHECK DRVLST FOR OTHER AVAILABLE DRIVES, AND RECALIBRATE THEM
022672  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
022700  005001                          CLR     R1              ;INIT MULTIPLE-DRIVES FLAG
022702  005000                          CLR     R0              ;INIT DRIVE NO. TO 0
022704  005065  000002                  CLR     P.CYLN(R5)      ;SET CYL = 0
022710  022700  000010          4$:     CMP     #10,R0          ;SEE IF ALL DRIVES CHECKED YET
022714  001415                          BEQ     8$              ;BR IF ALL DONE
022716  105760  005606                  TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
022722  001410                          BEQ     6$              ;BR IF NOT MARKED
022724  120037  005500                  CMPB    R0,DRIVE        ;SEE IF MARKED DRIVE IS DRIVE UNDER TEST
022730  001405                          BEQ     6$              ;BR IF YES
022732  110065  000000                  MOVB    R0,P.DRVN(R5)   ;SET DRIVE NO. PARAMETER
022736  004737  040774                  JSR     PC,DRVCAL       ;SEEK TO CYL 0 ON CURRENT DRIVE
022742  005201                          INC     R1              ;SET FLAG TO INDICATE MORE THAN 1 DRIVE
022744  005200          6$:     INC     R0                      ;INCR DRIVE NO.
022746  000760                  BR      4$                      ;KEEP CHECKING DRIVES
022750  005701          8$:     TST     R1                      ;SEE IF MORE THAN 1 DRIVE PRESENT
022752  001002                  BNE     9$                      ;BR IF YES, TO RUN TEST
022754  000137  023474          JMP     MULTI1                  ;NO, SKIP TEST
                                        ;SEEK TO CYLINDER FC ON DRIVE UNDER TEST, SET PARAMS FOR DATA XFER
022760  004737  030776          9$:     JSR     PC,INITSS       ;INIT S.S.
022764  013765  005646  000002          MOV     FC,P.CYLN(R5)   ;SET CYL = FC
022772  005065  000004                  CLR     P.SECT(R5)      ;SET TRACK AND SECTOR = 0
022776  012765  136400  000012          MOV     #-17152.,P.WC(R5) ;SET WC FOR 67(DEC) SECTORS
023004  105737  003115                  TSTB    FORMAT          ;CHECK THE FORMAT
023010  001403                          BEQ     12$             ;BR IF 22-SECTOR FORMAT
023012  012765  141400  000012          MOV     #-15616.,P.WC(R5) ;SET WC FOR 61(DEC) SECTORS
023020  012765  006664  000010  12$:    MOV     #PAT13,P.BALO(R5) ;SET BA BITS
023026  052765  100000  000014          BIS     #DTBAII,P.PRST(R5) ;SET BUS ADDRESS INCREMENT INHIBIT
023034  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
023042  004737  040774                  JSR     PC,DRVCAL       ;SEEK TO CYL FC ON DRIVE UNDER TEST
                                        ;DO PSEUDO-RANDOM SEEKS ON ALL OTHER DRIVES
023046  105037  003110                  CLRB    TSTING          ;INHIBIT ↑C, ↑Z ESCAPE
023052  012701  000007                  MOV     #7,R1           ;SET LOOP COUNTER
023056  012700  003202                  MOV     #CYLLST,R0      ;ADRS OF RAND CYL LIST
023062  004737  034122          15$:    JSR     PC,RNDADR       ;GENERATE A PSEUDO-RANDOM CYL NO.
023066  013720  005504                  MOV     CYLNDR,(R0)+    ;SET CYL NO. IN TABLE
023072  001002                          BNE     16$             ;BR IF NOT 0
023074  005260  177776                  INC     -2(R0)          ;MAKE IT NON-ZERO
023100  005301          16$:    DEC     R1                      ;DECREMENT LOOP COUNTER
023102  001367                  BNE     15$                     ;BR IF NOT DONE LOADING LIST YET
023104  012701  003202                  MOV     #CYLLST,R1      ;GET ADRS OF RAND CYL TABLE
023110  010537  023160                  MOV     R5,19$          ;SET PARAM BLK ADRS FOR DRIVER
023114  005000                          CLR     R0              ;INIT DRIVE NO. TO 0
023116  022700  000010          18$:    CMP     #10,R0          ;SEE IF ALL DRIVES CHECKED YET
```

```
023122  001421                            BEQ     22$               ;BR IF YES
023124  105760  005606                    TSTB    DRVLST(R0)        ;SEE IF THIS DRIVE PRESENT
023130  001414                            BEQ     20$               ;BR IF NO
023132  120037  005530                    CMPB    R0,DRIVE          ;SEE IF THIS IS DRIVE UNDER TEST
023136  001411                            BEQ     20$               ;BR IF YES
023140  110065  000000                    MOVB    R0,P.DRVN(R5)     ;SET DRIVE NO. PARAMETER
023144  012165  000002                    MOV     (R1)+,P.CYLN(R5)  ;SET CYLINDER NO.
023150  004737  041110                    JSR     PC,STRCMD         ;STORE PREV AND CURRENT CMNDS
023154  004737  051120                    JSR     PC,C.INIT         ;GO START SEEK ON THIS DRIVE
023160  000000                    19$:    .WORD                     ;P.B. ADRS GOES HERE
023162  005200                    20$:    INC     R0                ;INCR DRIVE NO.
023164  000754                            BR      18$               ;BR TO SEEK ON NEXT DRIVE
                                  ;WRITE DATA ON DRIVE UNDER TEST, WITH BAI SET
023166  112765  000123  000001    22$:    MOVB    #WRDATA,P.CMND(R5)  ;SET WRITE DATA COMMAND
023174  113765  005530  000000            MOVB    DRIVE,P.DRVN(R5)  ;SET DRIVE NO.
023202  013765  005646  000002            MOV     FC,P.CYLN(R5)     ;SET CYL NO.
023210  012737  023406  003034            MOV     #MULHDL,A.NORM    ;SET SPECIAL DRIVER RETURN ADDRESS
023216  004737  040774                    JSR     PC,DRVCAL         ;WRITE 1 CYL + 1 SECTOR AT CYL FC
023222  032737  000002  005474            BIT     #BSERR,RECODE     ;SEE IF BSE ERROR
023230  001121                            BNE     MULTI1            ;BR IF YES
023232  112765  000131  000001            MOVB    #WRTCHK,P.CMND(R5)  ;SET WRITE CHECK COMMAND
023240  004737  040774                    JSR     PC,DRVCAL         ;WRITE CHECK THE DATA WRITTEN
                                  ;CHECK ATTENTIONS AND CHECK CYLINDER ADDRESSES FROM ALL OTHER DRIVES
023244  112737  000001  003110            MOVB    #1,TSTING         ;ALLOW ↑C, ↑Z ESCAPE
023252  112765  000141  000001            MOVB    #RDSTAT,P.CMND(R5)  ;SET READ DRIVE STATUS COMMAND
023260  012701  003202                    MOV     #CYLLST,R1        ;ADRS OF RAND CYL LIST
023264  005000                            CLR     R0                ;INIT DRIVE NO.
023266  022700  000010            26$:    CMP     #10,R0            ;SEE IF ALL DRIVES CHECKED YET
023272  001500                            BEQ     MULTI1            ;BR IF YES
023274  105760  005606                    TSTB    DRVLST(R0)        ;SEE IF THIS DRIVE MARKED IN LIST
023300  001440                            BEQ     32$               ;BR IF NOT
023302  120037  005530                    CMPB    R0,DRIVE          ;SEE IF THIS IS DRIVE UNDER TEST
023306  001435                            BEQ     32$               ;BR IF YES
023310  136037  003072  003200            BITB    I.DRV(R0),SAVWRD  ;SEE IF GOT ATT'N FROM THIS DRIVE
023316  001003                            BNE     28$               ;BR IF YES
023320  004737  042206                    JSR     PC,REPSUP         ;PREPARE STATUS FOR PRINTOUT
023324  104113                            ERROR   113               ;"NO ATTENTION ON SEEK"
023326  110065  000000            28$:    MOVB    R0,P.DRVN(R5)     ;SET THIS DRIVE NO.
023332  004737  040774                    JSR     PC,DRVCAL         ;READ STATUS OF THIS DRIVE
023336  016503  000052                    MOV     P.B10(R5),R3      ;GET STATUS BYTE B10
023342  006003                            ROR     R3                ;GET CYL ADRS RIGHT-JUSTIFIED
023344  006003                            ROR     R3
023346  006003                            ROR     R3
023350  006003                            ROR     R3
023352  042703  177000                    BIC     #177000,R3        ;CLEAR OFF UNUSED BITS
023356  022103                            CMP     (R1)+,R3          ;COMPARE TO EXPECTED CYLINDER
023360  001410                            BEQ     32$               ;BR IF EQUAL
023362  016137  177776  001212            MOV     -2(R1),$REG14     ;GOOD CYL NO.
023370  010337  001214                    MOV     R3,$REG15         ;BAD CYL NO.
023374  004737  042206                    JSR     PC,REPSUP         ;GATHER STATUS FOR PRINTOUT
023400  104114                            ERROR   114               ;"DRIVE'S CYLINDER INCORRECT"
023402  005200                    32$:    INC     R0                ;INCREMENT DRIVE NO.
023404  000730                            BR      26$               ;BR IF NOT DONE WITH ALL DRIVES YET

                                  ;THIS IS THE NORMAL RETURN FROM THE DRIVER WHICH IS USED TO CHECK
                                  ;THE RESULTS OF MULTI-DRIVE OPERATIONS.
```

```
                                MULMOL: BIT     #RDY,T.CS1      ;SEE IF CNTRLR RDY IS SET
                                        BNE     6$              ;BR IF RDY SET
                                        JSR     PC,I.CSTS       ;READ CONTROLLER REGISTERS
                                        MOV     T.CS1,P.CS1(R5) ;GET CS1 BITS
                                        JSR     PC,REPSUF       ;PREPARE STATUS FOR PRINTOUT
                                        ERROR   112             ;"INTRPT WHEN CNTRLR NOT RDY"
                                6$:     MOV     T.ASOF,SAVWRD   ;SAVE ATT'N SUMMARY
                                        SWAB    SAVWRD          ;GET ATT'N BITS IN BITS 0-7
                                        MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
                                        MOV     #ERRFRE,A.NORM  ;RESTORE NORMAL DRIVER RETURN ADDRESS
                                        JSR     PC,DRVCAL       ;CLEAR SUB-SYS
                                        JMP     ERRFRE          ;TAKE NORMAL RETURN

                                MULTI1:


                                .SBTTL  END OF PASS ROUTINE

                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER,
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO NEWPAS

                                $EOP:
                                        SCOPE
                                        INC     $DEVCT          ;INCREMENT DEVICE COUNT FOR RPT
                                        JMP     NEWDRV          ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
                                DUNPAS: ;THIS IS TRULY THE END OF A PASS
                                        BIC     #BIT6,@$TKS     ;DISABLE TTY KBD INTERRUPT
                                        CLR     $TSTNM          ;;ZERO THE TEST NUMBER
                                        CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
                                        INC     $PASS           ;;INCREMENT THE PASS NUMBER
                                        BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
                                        DEC     (PC)+           ;;LOOP?
                                $EOPCT: .WORD   1
                                        BGT     $DOAGN          ;;YES
                                        MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
                                $ENDCT: .WORD   1
                                        $EOPCT
                                        TYPE    ,$ENDMG         ;;TYPE "END PASS #"
                                        MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                                        TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
                                $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
                                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
                                        RESET                   ;;CLEAR THE WORLD
                                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
                                        NOP                     ;;SAVE ROOM
                                        NOP                     ;;FOR
                                        NOP                     ;;ACT11
                                $DOAGN:
                                        JMP     @(PC)+          ;;RETURN
                                $RTNAD: .WORD   NEWPAS
                                $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
```

```
 6214  023617  000215  042412  042116  SENDMG: .ASCIZ  <15><12>/END PASS #/
 6215  023624  053040  051501  020123
 6216  023632  000043
 6217
 6218                                  .SBTTL  RK06 HEAD ALIGNMENT AID
```

```
  6219  023634  012737  000340  177776  ASTART: MOV     #PR7,@#PS
  6220                                  .SBTTL  INITIALIZE THE COMMON TAGS
  6221                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  6222  023642  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  6223  023646  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  6224  023650  022706  001140                  CMP     #SWR,R6 ;;DONE?
  6225  023654  001374                          BNE     .-6             ;;LOOP BACK IF NO
  6226  023656  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  6227                                  ;;INITIALIZE A FEW VECTORS
  6228  023662  012737  055504  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  6229  023670  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  6230  023676  012737  055000  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  6231  023704  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  6232  023712  012737  056612  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  6233  023720  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
  6234  023726  012737  055350  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  6235  023734  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  6236  023742  013737  023546  023540          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
  6237  023750  005037  001304                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  6238  023754  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  6239  023760  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  6240  023766  012737  023766  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  6241  023774  012737  023774  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
  6242                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  6243                                  ;;EQUAL TO A "-1"  SETUP FOR A SOFTWARE SWITCH REGISTER.
  6244  024002  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  6245  024006  012737  024042  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  6246  024014  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  6247  024022  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  6248  024030  022777  177777  155102          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  6249  024036  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  6250                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
  6251  024040  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
  6252  024042  012716  024050          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  6253  024046  000002                          RTI
  6254  024050  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  6255  024056  012737  000174  001142          MOV     #DISPREG,DISPLAY
  6256  024064  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  6257
  6258  024070  005037  001326                  CLR     $PASS           ;;CLEAR PASS COUNT
  6259  024074  132737  000200  001341          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  6260  024102  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
  6261  024104  012737  001342  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  6262  024112                          67$:
  6263                                  .SBTTL  RK06 HEAD ALIGNMENT AID
  6264  024112  000005                          RESET                   ;RESET UNIBUS
  6265  024114  104401  007024                  TYPE    ,DZR6N          ;TYPE PROGRAM I.D. FOR PART 2
  6266  024120  104401  007044                  TYPE    ,SUBVER
  6267  024124  104401  007125                  TYPE    ,PART2
  6268  024130  012737  000176  001140          MOV     #SWREG,SWR      ;ADRS OF SOFTWARE SWR
  6269  024136  005077  154776                  CLR     @SWR            ;MAKE SWR BITS ALL 0
  6270  024142  012737  027142  000060          MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
  6271  024150  012737  000200  000062          MOV     #PR4,@#TKVEC+2  ;SET kBD PRIORITY = 4
  6272  024156  013701  003030                  MOV     RKVEC,R1        ;GET ADDRESS OF VECTOR STORAGE
  6273  024162  012721  045620                  MOV     #I.INTR,(R1)+   ;SET IT TO INTERRUPT HNDLR
  6274  024166  012711  000340                  MOV     #PR7,(R1)       ;SET INTERRUPT HANDLER PR7
```

```
  6275  024172  012737  000000  177776          MOV     #PRO,@#PS       ;ALLOW ALL INTERRUPTS
  6276  024200  012737  000000  005646          MOV     #0,FC           ;SET FIRST CYL = 0
  6277  024206  012737  000632  005650          MOV     #LSTCYL,LC      ;SET LAST CYL = 410(10)
  6278
  6279  024214                          GIVEID:
  6280  024214  012737  176543  053246          MOV     #176543.,$HINUM ;INITIALIZE PSEUDO-RANDOM NUMBERS
  6281  024222  012737  123456  053250          MOV     #123456,$LONUM
  6282  024230  105037  003106                  CLRB    MDFLAG          ;CLEAR PARAM INP FLAG
  6283  024234  105037  003110                  CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
  6284  024240  005037  005502                  CLR     STALLS          ;DON'T STALL ON OPERATIONS
  6285  024244  012701  005222                  MOV     #PRVCMD,R1      ;ZERO OUT PREV COMMAND
  6286  024250  012700  000006                  MOV     #6,R0
  6287  024254  005021                  42$:    CLR     (R1)+
  6288  024256  005300                          DEC     R0
  6289  024260  001375                          BNE     42$
  6290  024262  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
  6291  024266  104401  011435                  TYPE    ,IDENT          ;TYPE PROGRAM IDENTIFICATION
  6292  024272  105737  003136                  TSTB    HLPOVL          ;SEE IF HELP FILE OVERLAID
  6293  024276  001011                          BNE     ASKMOD          ;BR IF YES
  6294  024300  104401  011502                  TYPE    ,FORHLP         ;ASK IF HELP DESIRED
  6295  024304  104406                          RDCHR                   ;READ RESPONSE
  6296  024306  112601                          MOVB    (SP)+,R1        ;GET CHARACTER
  6297  024310  122701  000015                  CMPB    #015,R1         ;SEE IF <CR> TYPED
  6298  024314  001402                          BEQ     ASKMOD          ;BR IF NO HELP NEEDED
  6299  024316  104401  065257                  TYPE    ,HLPFIL         ;HELP REQUESTED- TYPE INFO.
  6300
  6301
  6302                                  ;DETERMINE DESIRED OPERATIONAL MODE
  6303  024322  104401  011537  ASKMOD: TYPE    ,TYPMOD         ;ASK FOR DESIRED OPERATIONAL MODE
  6304  024326  004737  027302          JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
  6305  024332  005737  005522  1$:     TST     INTCHR          ;CHECK FOR TTY INPUT
  6306  024336  001775                  BEQ     1$              ;BR IF NO INPUT YET
  6307  024340  013701  005522          MOV     INTCHR,R1       ;GET INPUT CHAR INTO R1
  6308  024344  022701  000022          CMP     #022,R1         ;SEE IF (↑R) TYPED
  6309  024350  001721                  BEQ     GIVEID          ;BR IF (↑R), TO RESTART
  6310  024352  022701  000115          CMP     #'M,R1          ;IS IT MANUAL MODE ?
  6311  024356  001002                  BNE     2$              ;BR IF NOT MANUAL
  6312  024360  000137  024404          JMP     MANUAL          ;JUMP TO MANUAL MODE ROUTINE
  6313  024364  022701  000101  2$:     CMP     #'A,R1          ;IS IT AUTO MODE ?
  6314  024370  001002                  BNE     3$              ;BR IF  NOT AUTO
  6315  024372  000137  025332          JMP     AUTO            ;JUMP TO AUTO MODE ROUTINE
  6316  024376  004737  027322  3$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
  6317  024402  000747                  BR      ASKMOD          ;BR TO  ASK  FOR MODE AGAIN
  6318
  6319
  6320
  6321                                  ;MANUAL SELECT MODE ROUTINE
  6322  024404                  MANUAL:
  6323  024404  104401  011600          TYPE    ,TYPMAN         ;TYPE "MANUAL SELECT MODE"
  6324  024410  104401  011633  ASKDRV: TYPE    ,ENTDRV         ;ASK FOR DRIVE  NUMBER
  6325  024414  004737  027302          JSR     PC,PREPKB       ;PREPARE FOR KBD  INPUT
  6326  024420  005737  005522  1$:     TST     INTCHR          ;SEE IF ANY INPUT
  6327  024424  001775                  BEQ     1$              ;BR IF NONE TO CHECK AGAIN
  6328  024426  013701  005522          MOV     INTCHR,R1       ;GET CHARACTER INTO R1
  6329  024432  020127  000022          CMP     R1,#022         ;TEST FOR  (↑R) TYFED
  6330  024436  001666                  BEQ     GIVEID          ;BR IF (↑R) TO RESTART
```

```
6331  024440  020127  000060          CMP     R1,#'0          ;COMPARE TO ASCII  0
6332  024444  002410                  BLT     2$              ;BR IF <0 (BAD INPUT)
6333  024446  020127  000067          CMP     R1,#'7          ;COMPARE TO ASCII  7
6334  024452  003005                  BGT     2$              ;BR IF  >7 (BAD INPUT)
6335  024454  042701  000060          BIC     #'0,R1          ;STRIP  ASCII BITS
6336  024460  010137  005500          MOV     R1,DRIVE        ;STORE DRIVE NUMBER
6337  024464  000403                  BR      3$              ;PROCEED WITH DESIRED DRIVE
6338  024466  004737  027322  2$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6339  024472  000746                  BR      ASKDRV          ;GO BACK TO ASK AGAIN
6340  024474  004737  030776  3$:     JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
6341                          ;CHECK DESIRED DRIVE FOR PROPER STATUS
6342  024500  004737  031530          JSR     PC,CHKDRV       ;CHECK DRIVE FOR ON-LINE,READY,WRITE LOCK
6343  024504  024410                  ASKDRV                  ;ADDRESS OF ERROR RETURN FOR CHKDRV
6344  024506  004737  033602          JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
6345                          ;SET UP DESIRED SUB-MODE OF OPERATION
6346  024512                  SUBMOD:
6347  024512  105037  011434          CLRB    VERIFY          ;CLEAR VERIFY MODE FLAG
6348  024516  104401  012416          TYPE    ,ENTMOD         ;ASK FOR ALIGN OR EXERCISE SUB-MODE
6349  024522  004737  027302          JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
6350  024526  005737  005522  1$:     TST     INTCHR          ;SEE IF ANY INPUT
6351  024532  001775                  BEQ     1$              ;BR IF NONE TO CHECK AGAIN
6352  024534  013701  005522          MOV     INTCHR,R1       ;GET CHAR INTO R1
6353  024540  020127  000022          CMP     R1,#022         ;SEE IF (↑R) TYPED
6354  024544  001002                  BNE     2$              ;BR IF NOT (↑R)
6355  024546  000137  024214          JMP     GIVEID          ;JUMP TO RESTART
6356  024552  020127  000003  2$:     CMP     R1,#003         ;SEE IF (↑C) TYPED
6357  024556  001002                  BNE     3$              ;BR IF NOT (↑C)
6358  024560  000137  024410          JMP     ASKDRV          ;JUMP TO ASK FOR DRIVE AGAIN
6359  024564  020127  000101  3$:     CMP     R1,#'A          ;SEE IF ALIGNMENT REQUESTED
6360  024570  001415                  BEQ     MANAL           ;BR IF MANUAL ALIGNMENT REQUESTED
6361  024572  020127  000105          CMP     R1,#'E          ;SEE IF EXERCISES REQUESTED
6362  024576  001002                  BNE     4$              ;BR IF NOT EXERCISES
6363  024600  000137  025076          JMP     MANEX           ;JUMP TO DO MANUAL EXERCISES
6364  024604  020127  000126  4$:     CMP     R1,#'V          ;SEE IF VERIFY REQUESTED
6365  024610  001002                  BNE     6$              ;BR IF NOT VERIFY
6366  024612  000137  025060          JMP     MANVR           ;JUMP TO VERIFY MODE
6367  024616  004737  027322  6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6368  024622  000733                  BR      SUBMOD          ;ASK FOR SUB-MODE AGAIN
6369
6370
6371
6372                          ;MANUAL ALIGNMENT MODE
6373  024624                  MANAL:
6374  024624  104401  012556          TYPE    ,MANALN         ;REPORT MANUAL ALIGNMENT MODE
6375                          ;REQUEST AND CHECK DESIRED HEAD
6376  024630  104401  012724  ASKHED: TYPE    ,ENTHED         ;ASK FOR DESIRED HEAD
6377  024634  004737  027302          JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
6378  024640  005737  005522  1$:     TST     INTCHR          ;SEE IF ANY INPUT
6379  024644  001775                  BEQ     1$              ;BR IF NONE TO CHECK AGAIN
6380  024646  013701  005522          MOV     INTCHR,R1       ;GET CHAR INTO R1
6381  024652  020127  000022          CMP     R1,#022         ;SEE IF (↑R)
6382  024656  001007                  BNE     2$              ;BR IF NOT (↑R)
6383  024660  112765  000113  000001  MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6384  024666  004737  040774          JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6385  024672  000137  024214          JMP     GIVEID          ;JUMP TO RESTART
6386  024676  020127  000003  2$:     CMP     R1,#003         ;SEE IF (↑C)
```

```
 6387  024702  001007                          BNE     3$              ;BR IF NOT (↑C)
 6388  024704  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6389  024712  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6390  024716  000137  024410                  JMP     ASKDRV          ;JUMP TO ASK FOR DRIVE AGAIN
 6391  024722  020127  000032          3$:     CMP     R1,#032         ;SEE IF (↑Z)
 6392  024726  001007                          BNE     4$              ;BR IF NOT (↑Z)
 6393  024730  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6394  024736  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6395  024742  000137  024512                  JMP     SUBMOD          ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
 6396  024746  020127  000060          4$:     CMP     R1,#'0          ;COMPARE TRACK TO ASCII 0
 6397  024752  002410                          BLT     5$              ;BR IF <0 (BAD INPUT)
 6398  024754  020127  000062                  CMP     R1,#'2          ;COMPARE TRACK TO ASCII 2
 6399  024760  003005                          BGT     5$              ;BR IF >2 (BAD INPUT)
 6400  024762  042701  000060                  BIC     #'0,R1          ;STRIP ASCII BITS
 6401  024766  010137  005520                  MOV     R1,TRACK        ;STORE TRACK (HEAD) NUMBER
 6402  024772  000404                          BR      6$              ;GO SELECT HEAD
 6403  024774  004737  027322          5$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6404  025000  000137  024630                  JMP     ASKHED          ;ASK AGAIN FOR HEAD NUMBER
 6405                                  ;UNLOAD HEADS,AND WHEN READY, START SPINDLE AND SEEK TO ALN CYL
 6406  025004  105737  011434          6$:     TSTB    VERIFY          ;SEE IF VERIFY MODE
 6407  025010  001002                          BNE     8$              ;BR IF VERIFY
 6408  025012  004737  032024                  JSR     PC,WAIT4R       ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
 6409  025016  113765  005520  000005  8$:     MOVB    TRACK,P.TRCK(R5) ;SET DESIRED HEAD NUMBER
 6410  025024  004737  031770                  JSR     PC,ALNSEK       ;SEEK TO ALIGNMENT CYLINDER
 6411  025030  104401  012653                  TYPE    ,HEDPOS         ;TYPE HEADS POSITIONED MSG
 6412  025034  013701  005520                  MOV     TRACK,R1        ;GET BINARY TRACK NO.
 6413  025040  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
 6414  025044  110137  013C12                  MOVB    R1,HEADNO       ;GET HEAD NO. INTO MSG BUFF.
 6415  025050  104401  013005                  TYPE    ,HEDSEL         ;TYPE HEAD SELECTED MSG
 6416  025054  000137  024630                  JMP     ASKHED          ;GO BACK TO ASK FOR NEW HEAD SELECT
 6417
 6418
 6419
 6420                                  ;MANUAL SELECT VERIFY MODE
 6421  025060  112737  000001  011434  MANVR:  MOVB    #1,VERIFY       ;SET VERIFY MODE FLAG
 6422  025066  104401  012616                  TYPE    ,MANVRF         ;REPORT MANUAL VERIFY MODE
 6423  025072  000137  024630                  JMP     ASKHED          ;GO ASK FOR DESIRED HEAD
 6424
 6425
 6426
 6427                                  ;MANUAL RANDOM SEEK EXERCISE ROUTINE
 6428  025076                          MANEX:
 6429  025076  004737  032024                  JSR     PC,WAIT4R       ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
 6430  025102  013701  005500                  MOV     DRIVE,R1        ;GET DRIVE NUMBER
 6431  025106  052701  000060                  BIS     #'0,R1          ;CONVERT TO ASCII
 6432  025112  110137  012550                  MOVB    R1,DRIEXR       ;PUT DRIVE NUMBER INTO OUTPUT BUFFER
 6433  025116  104401  012471                  TYPE    ,MANEXR         ;TYPE RANDOM SEEKS MSG
 6434  025122  012700  016514                  MOV     #RNDLNG,R0      ;INITIALIZE RANDOM SEEK COUNTER
 6435  025126  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 6436  025134  004737  027302          1$:     JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 6437  025140  004737  034122          2$:     JSR     PC,RNDADR       ;SELECT RANDOM CYLINDER ADDRESS
 6438  025144  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
 6439  025152  004737  040774                  JSR     PC,DRVCAL       ;DO A RANDOM SEEK
 6440                                  ;CHECK FOR TTY INPUT DURING EXERCISES
 6441  025156  005737  005522                  TST     INTCHR          ;SEE IF ANY INPUT
 6442  025162  001016                          BNE     3$              ;BR IF A CHAR WAS TYPED
```

```
6442  025164  005300                          DEC     R0              ;DECREMENT SEEK COUNTER
6444  025166  001364                          BNE     2$              ;BR IF NOT DONE SEEKING YET
6445  025170  042777  000100  153746          BIC     #BIT6,@#TKS     ;DISABLE KBD INTERRUPT
6446  025176  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6447  025204  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6448  025210  104401  001310                  TYPE    ,$BELL          ;RING BELL TO SIGNIFY END OF EXERCISES
6449  025214  000137  024512                  JMP     @SUBMOD         ;GO ASK FOR NEW MANUAL SUB-MODE
6450  025220  013701  005522          3$:     MOV     INTCHR,R1       ;GET CHARACTER INTO R1
6451  025224  020127  000022                  CMP     R1,#022         ;SEE IF (↑R)
6452  025230  001007                          BNE     4$              ;BR IF NOT (↑R)
6453  025232  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6454  025240  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6455  025244  000137  024214                  JMP     GIVEID          ;JUMP TO RESTART
6456  025250  020127  000003          4$:     CMP     R1,#003         ;SEE IF (↑C)
6457  025254  001007                          BNE     5$              ;BR IF NOT (↑C)
6458  025256  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6459  025264  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6460  025270  000137  024410                  JMP     ASKDRV          ;JUMP TO ASK FOR NEW DRIVE NUMBER
6461  025274  020127  000032          5$:     CMP     R1,#032         ;SEE IF (↑Z)
6462  025300  001007                          BNE     6$              ;BR IF NOT (↑Z)
6463  025302  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6464  025310  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6465  025314  000137  024512                  JMP     SUBMOD          ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
6466  025320  004737  027322          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6467  025324  104401  012471                  TYPE    ,MANEXR         ;VERIFY STILL IN MANUAL EXERCISES
6468  025330  000701                          BR      1$              ;BR TO CONTINUE EXERCISES
6469
6470
6471
6472                                  ;AUTO SELECT MODE ROUTINE
6473  025332                          AUTO:
6474  025332  004737  030776                  JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
6475  025336  104401  013027                  TYPE    ,TYPAUT         ;TYPE "AUTO SELECT MODE"
6476  025342  105037  011434          1$:     CLRB    @VERIFY         ;CLEAR VERIFY MODE FLAG
6477  025346  104401  012416                  TYPE    ,ENTMOD         ;ASK FOR ALIGN,VERIFY, OR EXERCISE
6478  025352  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR KEYBOARD INPUT
6479  025356  005737  005522          2$:     TST     @INTCHR         ;SEE IF ANY INPUT
6480  025362  001775                          BEQ     2$              ;BR IF NONE, TO CHECK AGAIN
6481  025364  013701  005522                  MOV     INTCHR,R1       ;GET CHAR INTO R1
6482  025370  020127  000022                  CMP     R1,#022         ;SEE IF (↑R) TYPED
6483  025374  001002                          BNE     3$              ;BR IF NOT (↑R)
6484  025376  000137  024214                  JMP     GIVEID          ;JUMP TO RESTART
6485  025402  020127  000101          3$:     CMP     R1,#'A          ;SEE IF AUTO ALIGNMENT REQUESTED
6486  025406  001417                          BEQ     AUTAL           ;BR IF AUTO ALIGNMENT REQUESTED
6487  025410  020127  000105                  CMP     R1,#'E          ;SEE IF AUTO EXERCISES REQUESTED
6488  025414  001002                          BNE     4$              ;BR IF EXERCISES NOT REQUESTED
6489  025416  000137  026442                  JMP     AUTEX           ;JUMP TO DO AUTO EXERCISES
6490  025422  020127  000126          4$:     CMP     R1,#'V          ;SEE IF VERIFY REQUESTED
6491  025426  001004                          BNE     6$              ;BR IF NOT VERIFY
6492  025430  112737  000001  011434          MOVB    #1,@VERIFY      ;SET VERIFY MODE FLAG
6493  025436  000403                          BR      AUTAL           ;PROCEED IN VERIFY MODE
6494  025440  004737  027322          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6495  025444  000736                          BR      1$              ;BR TO ASK FOR A OR E AGAIN
6496
6497
6498
```

```
                                        ;AUTO ALIGNMENT MODE
025446                                  AUTAL:
025446  105737  011434                        TSTB    VERIFY          ;SEE IF AUTO VERIFY MODE
025452  001403                                BEQ     3$              ;BR IF NOT
025454  104401  013116                        TYPE    ,AUTVRF         ;REPORT AUTO VERIFY MODE
025460  000402                                BR      4$
025462  104401  013060                3$:     TYPE    ,AUTALN         ;TYPE ALIGNMENT MESSAGE
025466  004737  027302                4$:     JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
025472  005037  003014                        CLR     T.MR2           ;CLEAR MR2 STORAGE WORD
025476  005037  005524                        CLR     SELECT          ;INITIALIZE AUTO-SELECTED DRIVE INDICATOR
025502  012737  000377  005526                MOV     #377,ONLINE     ;INIT. OLD ONLINE DRIVE INDICATOR
025510  012737  000377  005530                MOV     #377,NEWON      ;INIT. NEW ONLINE DRIVE INDICATOR
025516  012737  000377  005532                MOV     #377,SCRACH     ;INITIALIZE LAST DRIVE INDICATOR
025524  005737  005522            DRVLJP:     TST     INTCHR          ;SEE IF ANY KBD INPUT
025530  001445                                BEQ     6$              ;BR IF NO INPUT
025532  113765  005532  000000                MOVB    SCRACH,P.DRVN(R5) ;DRIVE NO. FOR POSSIBLE RECALIBRATE
025540  123727  005522  000032                CMPB    INTCHR,#032     ;SEE IF (↑Z) TYPED
025546  001013                                BNE     2$              ;BR IF NOT (↑Z)
025550  022737  000377  005532                CMP     #377,SCRACH     ;SEE IF ANY DRIVE SELECTED YET
025556  001405                                BEQ     1$              ;BR IF NONE
025560  112765  000113  000001                MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
025566  004737  040774                        JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
025572  000137  025332            1$:         JMP     AUTO            ;RESTART AUTO MODE
025576  123727  005522  000022    2$:         CMPB    INTCHR,#022     ;SEE IF (↑R) TYPED
025604  001013                                BNE     4$              ;BR IF NOT (↑R)
025606  022737  000377  005532                CMP     #377,SCRACH     ;SEE IF ANY DRIVE SELECTED YET
025614  001405                                BEQ     3$              ;BR IF NONE
025616  112765  000113  000001                MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
025624  004737  040774                        JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
025630  000137  024214            3$:         JMP     GIVEID          ;RESTART ALIGNMENT AID
025634  004737  027322            4$:         JSR     PC,ECOBAD       ;ECHO BAD INPUT
025640  004737  027302                        JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE  KBD INPUT
025644  005037  005500            6$:         CLR     DRIVE           ;ZERO THE DRIVE NUMBER
025650  012700  000001                        MOV     #1,R0           ;INITIALIZE DRIVE BIT MASK
025654  012737  027064  003036    8$:         MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
025662  050037  005530                        BIS     R0,NEWON        ;SET ON-LINE BIT FOR THIS DRIVE
                                        ;SELECT CURRENT DRIVE, CHECK FOR NON-EXISTENT DRIVE (NED) INDICATION
025666  113765  005500  000000                MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
025674  112765  000101  000001                MOVB    #SELDRV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
025702  004737  040774                        JSR     PC,DRVCAL       ;SELECT THIS DRIVE
025706  022737  000007  005500                CMP     #7,DRIVE        ;SEE IF WE JUST CHECKED DRIVE 7
025714  001405                                BEQ     10$             ;BR IF IT WAS DRIVE 7
025716  005237  005500                        INC     DRIVE           ;ADD 1 TO  DRIVE NUMBER
025722  000241                                CLC                     ;CLEAR CARRY BEFORE ROTATE
025724  006100                                ROL     R0              ;SHIFT BIT POINTER
025726  000752                                BR      8$              ;BR TO SELECT NEXT DRIVE
025730  013701  005530            10$:        MOV     NEWON,R1
                                        ;****** TO BE REMOVED *******
025734  023737  005530  005526                CMP     NEWON,ONLINE
025742  001670                                BEQ     DRVLUP
025744  012703  000001                        MOV     #1,R3           ;NUMBER OF 200 MILLI-SEC STALLS
025750  012700  177777            LOOP1:      MOV     #177777,R0
025754  005300                    LOOP2:      DEC     R0
025756  005700                                TST     R0
025760  001375                                BNE     LOOP2
025762  005303                                DEC     R3
```

```
  6555  025764  001371                          BNE     LOOP1
  6556
  6557  025766  043701  005526                  BIC     ONLINE,R1       ;GET CHANGED ONLINE BITS
  6558  025772  013737  005530  005526          MOV     NEWON,ONLINE    ;UPDATE ONLINE DRIVE BITS
  6559  026000  005701                          TST     R1              ;SEE IF ANY DRIVE JUST WENT ON-LINE
  6560  026002  001650                          BEQ     DRVLUP          ;BR IF NO DRIVE JUST WENT ONLINE
  6561                                  ;SERVICE THE DRIVE WHICH WAS JUST SELECTED
  6562  026004  012737  042466  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE USUAL  ERROR HANDLER ADDRESS
  6563  026012  005037  005500                  CLR     DRIVE           ;INITIALIZE DRIVE NO.
  6564  026016  010103                          MOV     R1,R3           ;COPY NEW SELECTED DRIVE BITS
  6565  026020  012700  000001                  MOV     #1,R0           ;INITIALIZE BIT POINTER
  6566  026024  030003                  12$:    BIT     R0,R3           ;SEE IF THIS BIT  IS  SET
  6567  026026  001005                          BNE     14$             ;BR IF THIS BIT SET
  6569  026030  005237  005500                  INC     DRIVE           ;INCREMENT DRIVE NO.
  6569  026034  000241                          CLC                     ;CLEAR CARRY BEFORE ROTATE
  6570  026036  006100                          ROL     R0              ;SHIFT BIT POINTER
  6571  026040  000771                          BR      12$             ;TRY AGAIN
  6572  026042  040003                  14$:    BIC     R0,R3           ;CLEAR OUT THIS BIT
  6573  026044  001415                          BEQ     16$             ;BR IF ONLY ONE DRIVE SELECTED
  6574                                  ;ERROR  -  MORE THAN ONE DRIVE SELECTED SIMULTANEOUSLY
  6575  026046  104401  012317                  TYPE    ,MULSE          ;REPORT ERROR
  6576  026052  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;INHIBIT RK06 INTERRUPT
  6577  026060  000000                          HALT                    ;HALT  FOR  MANUAL  INTERVENTION
  6578                                          ;       PRESS 'CONT' TO  PROCEED
  6579  026062  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
  6580  026070  004737  040774                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
  6581  026074  000137  025446                  JMP     AUTAL           ;RESTART  AUTO ALIGNMENT
  6582                                  ;CONTINUE WITH SELECTED DRIVE
  6583  026100  020137  005524          16$:    CMP     R1,SELECT       ;SEE IF  STILL  ALIGNING SAME  DRIVE
  6584  026104  001506                          BEQ     26$             ;BR IF SAME DRIVE
  6585                                  ;PROCEED WITH NEWLY-SELECTED DRIVE
  6586  026106  022737  000377  005532          CMP     #377,SCRACH     ;SEE IF THIS IS FIRST DRIVE SELECTED
  6587  026114  001421                          BEQ     23$             ;BR IF FIRST DRIVE
  6588  026116  113765  005532  000000          MOVB    SCRACH,P.DRVN(R5) ;GET LAST DRIVE NUMBER
  6589  026124  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
  6590  026132  004737  040774                  JSR     PC,DRVCAL       ;READ STATUS OF PREVIOUS DRIVE
  6591  026136  032765  000040  000044          BIT     #S.HDHM,P.AD1(R5) ;SEE IF HEADS ARE UNLOADED
  6592  026144  001405                          BEQ     23$             ;BR IF HEADS NOT UNLOADED
  6593  026146  112765  000111  000001          MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
  6594  026154  004737  040774                  JSR     PC,DRVCAL       ;START SPINDLE ON PREVIOUS DRIVE
  6595  026160  113737  005500  005532  23$:    MOVB    DRIVE,SCRACH    ;STORE DRIVE NO. FOR LATER CLEANUP STRT SPL
  6596  026166  010137  005524                  MOV     R1,SELECT       ;UPDATE SELECTED DRIVE NUMBER
  6597  026172  013701  005500                  MOV     DRIVE,R1        ;GET DRIVE NUMBER
  6598  026176  052701  000060                  BIS     #'0,R1          ;CONVERT TO ASCII
  6599  026202  110137  013160                  MOVB    R1,DRVSEL       ;GET DRIVE NUMBER INTO OUTPUT BUFFER
  6600  026206  104401  013151                  TYPE    ,DRISEL         ;TYPE "DRIVE X SELECTED"
  6601  026212  004737  031530                  JSR     PC,CHKDRV       ;CHECK DRIVE FOR RDY, WRITE PROT.
  6602  026216  025446                          AUTAL                   ;ERROR RETURN ADDRESS
  6603  026220  112737  000377  011433          MOVB    #377,UNLOD      ;SET DRIVE UNLOADED INDICATOR
  6604  026226  012737  000002  005520          MOV     #2,TRACK        ;SET TRACK = 2
  6605  026234  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
  6606  026242  105737  011434          24$:    TSTB    VERIFY          ;SEE IF AUTO VERIFY
  6607  026246  001016                          BNE     32$             ;SKIP UNLOAD IF VERIFY
  6608  026250  112765  000107  000001          MOVB    #UNLOAD,P.CMND(R5) ;SET UNLOAD COMMAND
  6609  026256  004737  040774                  JSR     PC,DRVCAL       ;UNLOAD THE DRIVE
  6610  026262  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
```

```
6611  026270  004737  040774          25$:    JSR     PC,DRVCAL       ;READ STATUS OF DRIVE
6612  026274  032765  000040  000044          BIT     #S.HDHM,P.AD1(R5) ;SEE IF HEADS ARE UNLODED YET
6613  026302  001772                          BEQ     25$             ;BR IF NOT YET
6614  026304  112765  000177  000001  32$:    MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6615  026312  004737  040774                  JSR     PC,DRVCAL       ;CLEAR THE S.S.
6616  026316  000137  025524                  JMP     DRVLUP          ;GO BACK TO SCAN DRIVES AGAIN
6617                                  ;PROCEED WITH SAME DRIVE
6618  026322  113765  005500  000000  26$:    MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6619  026330  105137  011433                  COMB    UNLOD           ;COMPLEMENT UNLOAD INDICATOR
6620  026334  001342                          BNE     24$             ;BR IF DRIVE SHOULD BE UNLOADED
6621  026336  105737  011434                  TSTB    VERIFY          ;SEE IF AUTO VERIFY
6622  026342  001005                          BNE     33$             ;SKIP START SPL IF VERIFY
6623  026344  112765  000111  000001          MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
6624  026352  004737  040774                  JSR     PC,DRVCAL       ;START SPINDLE ON THIS DRIVE
6625  026356  022737  000002  005520  33$:    CMP     #2,TRACK        ;SEE IF LAST TRACK WAS 2
6626  026364  001003                          BNE     28$             ;BR IF IT WAS NOT 2
6627  026366  005037  005520                  CLR     TRACK           ;SET TRACK = 0
6628  026372  000402                          BR      30$             ;GO SEEK TO ALIGNMENT CYLINDER
6629  026374  005237  005520          28$:    INC     TRACK           ;INCREMENT TRACK NUMBER
6630  026400  113765  005520  000005  30$:    MOVB    TRACK,P.TRCK(R5) ;SET TRACK NO.
6631  026406  004737  031770                  JSR     PC,ALNSEK       ;SEEK IN INCREMENTS TO ALIGN. CYL
6632  026412  104401  012653                  TYPE    ,HEDPOS         ;TYPE HEADS POSITIONED MSG
6633  026416  013701  005520                  MOV     TRACK,R1        ;GET CURRENT TRACK NO.
6634  026422  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
6635  026426  110137  013012                  MOVB    R1,HEADNO       ;GET TRACK INTO MSG BUFFER
6636  026432  104401  013005                  TYPE    ,HEDSEL         ;TYPE "HEAD X SELECTED"
6637  026436  000137  025524                  JMP     DRVLUP          ;GO BACK TO SCAN DRIVES AGAIN
6638
6639
6640                                  ;AUTO RANDOM SEEK EXERCISE ROUTINE
6641  026442                          AUTEX:
6642  026442  004737  030776                  JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
6643  026446  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
6644  026452  104401  013176                  TYPE    ,AUTEXR         ;ASK FOR SHORT OR LONG SEEK EXERCISES
6645  026456  005737  005522          1$:     TST     INTCHR          ;SEE IF ANY INPUT
6646  026462  001775                          BEQ     1$              ;BR IF NO INPUT
6647  026464  023727  005522  000022          CMP     INTCHR,#022     ;SEE IF (↑R) TYPED
6648  026472  001002                          BNE     2$              ;BR IF NOT (↑R)
6649  026474  000137  024214                  JMP     GIVEID          ;JUMP TO RESTART ALIGNMENT AID
6650  026500  023727  005522  000032  2$:     CMP     INTCHR,#032     ;SEE IF (↑Z) TYPED
6651  026506  001002                          BNE     3$              ;BR IF NOT (↑Z)
6652  026510  000137  025332                  JMP     AUTO            ;JUMP TO RESTART AUTO MODE
6653  026514  023727  005522  000123  3$:     CMP     INTCHR,#'S      ;SEE IF SHORT EXERCISES DESIRED
6654  026522  001004                          BNE     4$              ;BR IF NOT SHORT
6655  026524  012737  002734  005532          MOV     #RNDSHT,SCRACH  ;SET 1 MINUTE SEEK COUNT
6656  026532  000413                          BR      6$              ;PROCEED WITH SHORT EXERCISES
6657  026534  023727  005522  000114  4$:     CMP     INTCHR,#'L      ;SEE IF LONG EXERCISES DESIRED
6658  026542  001004                          BNE     5$              ;BR IF NOT LONG
6659  026544  012737  016514  005532          MOV     #RNDLNG,SCRACH  ;SET 5 MINUTE SEEK COUNT
6660  026552  000403                          BR      6$              ;PROCEED WITH LONG EXERCISES
6661  026554  004737  027322          5$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6662  026560  000730                          BR      AUTEX           ;BR TO ASK AGAIN
6663  026562  005037  005500          6$:     CLR     DRIVE           ;SET DRIVE = 0
6664                                  ;SELECT CURRENT DRIVE AND CHECK FOR NON-EXISTENT DRIVE INDICATION
6665  026566  012737  027064  003036  8$:     MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
6666  026574  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
```

```
6667  026602  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
6668  026610  012737  000377  005530          MOV     #377,NEWON         ;INITIALIZE ON-LINE INDICATOR
6669  026616  004737  040774                  JSR     PC,DRVCAL          ;SELECT THIS DRIVE
6670  026622  012737  042466  003036          MOV     #ERRHDL,A.ABNL     ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
6671  026630  022737  000377  005530          CMP     #377,NEWON         ;SEE IF NED INDICATION ON THIS DRIVE
6672  026636  001075                          BNE     18$                ;BR IF NED, TO SKIP THIS DRIVE
6673                                  ;CHECK DRIVE FOR RDY, WRITE PROTECT
6674  026640  004737  031530                  JSR     PC,CHKDRV          ;CHECK THIS DRIVE
6675  026644  026442                          AUTEX                      ;ERROR RETURN ADDRESS FOR CHKDRV
6676                                  ;PROCEED WITH SEEK EXERCISES ON THIS DRIVE
6677  026646  013701  005500                  MOV     DRIVE,R1           ;GET DRIVE # IBER
6678  026652  052701  000060                  BIS     #'0,R1             ;CONVERT TO ASCII
6679  026656  110137  013310                  MOVB    R1,AUTODR          ;GET DRIVE NUMBER INTO OUTPUT BUFFER
6680  026662  104401  013267                  TYPE    ,AUTOEX            ;TYPE "EXERCISING DRIVE X"
6681  026666  112765  000117  000001          MOVB    #SEEK,P.CMND(R5)   ;SET SEEK COMMAND
6682  026674  013700  005532                  MOV     SCRACH,R0          ;SEEK COUNT
6683  026700  004737  027302          10$:    JSR     PC,PREPKB          ;PREPARE FOR POSSIBLE KBD INPUT
6684  026704  004737  034122                  JSR     PC,RNDADR          ;SELECT RANDOM CYLINDER ADDRESS
6685  026710  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5)  ;SET RANDOM CYLINDER ADDRESS
6686  026716  004737  040774                  JSR     PC,DRVCAL          ;DO A RANDOM SEEK
6687  026722  005737  005522                  TST     INTCHR             ;SEE IF ANY KBD INPUT
6688  026726  001432                          BEQ     16$                ;BR IF NO INPUT
6689  026730  023727  005522  000022          CMP     INTCHR,#022        ;SEE IF (↑R) TYPED
6690  026736  001007                          BNE     12$                ;BR IF NOT (↑R)
6691  026740  112765  000113  000001          MOVB    #RECAL,P.CMND(R5)  ;SET RECALIBRATE COMMAND
6692  026746  004737  040774                  JSR     PC,DRVCAL          ;RECALIBRATE DRIVE
6693  026752  000137  024214                  JMP     GIVEID             ;JUMP TO RESTART ALIGNMENT AID
6694  026756  023727  005522  000032  12$:    CMP     INTCHR,#032        ;SEE IF (↑Z) TYPED
6695  026764  001007                          BNE     14$                ;BR IF NOT (↑Z)
6696  026766  112765  000113  000001          MOVB    #RECAL,P.CMND(R5)  ;SET RECALIBRATE COMMAND
6697  026774  004737  040774                  JSR     PC,DRVCAL          ;RECALIBRATE DRIVE
6698  027000  000137  025332                  JMP     AUTO               ;JUMP TO RESTART AUTO MODE
6699  027004  004737  027322          14$:    JSR     PC,ECOBAD          ;ECHO BAD INPUT
6700  027010  004737  027302                  JSR     PC,PREPKB          ;ENABLE KBD INPUT AGAIN
6701  027014  005300                  16$:    DEC     R0                 ;DECREMENT SEEK COUNT
6702  027016  001330                          BNE     10$                ;BR IF NOT DONE WITH THIS DRIVE
6703  027020  112765  000113  000001          MOVB    #RECAL,P.CMND(R5)  ;SET RECALIBRATE COMMAND
6704  027026  004737  040774                  JSR     PC,DRVCAL          ;RECALIBRATE DRIVE
6705  027032  005237  005500          18$:    INC     DRIVE              ;INCREMENT DRIVE NUMBER
6706  027036  022737  000010  005500          CMP     #10,DRIVE          ;SEE IF DONE WITH ALL DRIVES
6707  027044  001250                          BNE     8$                 ;BR IF MORE DRIVES TO EXERCISE YET
6708  027046  042777  000100  152070          BIC     #BIT6,@$TKS        ;DISABLE KBD INTERRUPT
6709  027054  104401  001310                  TYPE    ,$BELL             ;RING BELL AT END OF AUTO-EXERCISES
6710  027060  000137  025332                  JMP     AUTO               ;RESTART AUTO MODE
6711
6712
6713                                  ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
6714                                  ;  WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
6715                                  ;    (NED = NON-EXISTENT DRIVE)
6716  027064  032765  010000  000020  NEDHDL: BIT     #NED,P.CS2(R5)     ;SEE IF NED ON DRIVE SELECT
6717  027072  001002                          BNE     1$                 ;BR IF NED SET
6718  027074  000137  042466                  JMP     ERRHDL             ;GO HANDLE  OTHER ERROR
6719  027100  010046                  1$:     MOV     R0,-(SP)           ;SAVE R0,R1
6720  027102  010146                          MOV     R1,-(SP)
6721  027104  012700  000001                  MOV     #1,R0              ;SET BIT POINTER
6722  027110  005001                          CLR     R1                 ;CLEAR COUNTER
```

```
 6723  027112  120137  005500          2$:     CMPB    R1,DRIVE        ;SEE IF R1 = CURRENT DRIVE NUMBER
 6724  027116  001403                          BEQ     3$              ;BR IF  =
 6725  027120  005201                          INC     R1              ;INCREMENT COUNTER
 6726  027122  006300                          ASL     R0              ;SHIFT BIT POINTER
 6727  027124  000772                          BR      2$              ;TRY AGAIN
 6728  027126  040037  005530          3$:     BIC     R0,NEWON        ;CLEAR ONLINE BIT FOR  THIS DRIVE
 6729  027132  012601                          MOV     (SP)+,R1        ;RESTORE R0,R1
 6730  027134  012600                          MOV     (SP)+,R0
 6731  027136  000137  044702                  JMP     RETNML          ;TAKE  NORMAL  RETURN
 6732
 6733
 6734                                  ;;*****************************************************************
 6735                                  .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
 6736                                  ;;*****************************************************************
 6737
 6738  027142  010146                  KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
 6739  027144  017701  151776                  MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
 6740  027150  042701  177600                  BIC     #177600,R1      ;MASK OUT UNUSED BITS
 6741  027154  120127  000172                  CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
 6742  027160  003005                          BGT     20$             ;BR IF NOT
 6743  027162  120127  000141                  CMPB    R1,#141
 6744  027166  002402                          BLT     20$             ;BR IF NOT
 6745  027170  042701  000040                  BIC     #BIT5,R1        ;MAKE IT UPPER CASE
 6746  027174  010137  005522          20$:    MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
 6747  027200  122701  000003                  CMPB    #003,R1         ;SEE IF (↑C) TYPED
 6748  027204  001007                          BNE     2$              ;BR IF NOT (↑C)
 6749  027206  104401  013360                  TYPE    ,CNTRLC         ;ECHO (↑C)
 6750  027212  042777  000100  151724  1$:     BIC     #BIT6,@$TKS     ;CLEAR KBD INTERRUPT ENABLE BIT
 6751  027220  012601                          MOV     (SP)+,R1        ;RESTORE R1
 6752  027222  000002                          RTI                     ;RETURN
 6753  027224  122701  000032          2$:     CMPB    #032,R1         ;SEE IF (↑Z) TYPED
 6754  027230  001003                          BNE     3$              ;BR IF NOT (↑Z)
 6755  027232  104401  013365                  TYPE    ,CNTRLZ         ;ECHO (↑Z)
 6756  027236  000765                          BR      1$              ;RETURN
 6757  027240  122701  000022          3$:     CMPB    #022,R1         ;SEE IF (↑R) TYPED
 6758  027244  001003                          BNE     4$              ;BR IF NOT (↑R)
 6759  027246  104401  013372                  TYPE    ,CNTRLR         ;ECHO (↑R)
 6760  027252  000757                          BR      1$              ;RETURN
 6761  027254  122701  000007          4$:     CMPB    #007,R1         ;SEE IF (↑G) TYPED
 6762  027260  001003                          BNE     5$              ;BR IF NOT (↑G)
 6763  027262  104401  013404                  TYPE    ,CNTRLG         ;ECHO (↑G)
 6764  027266  000751                          BR      1$              ;RETURN
 6765  027270  104401  005522          5$:     TYPE    ,INTCHR         ;ECHO INPUT
 6766  027274  104401  001315                  TYPE    ,$CRLF          ;DO <CR> AND <LF>
 6767  027300  000744                          BR      1$              ;RETURN
 6768
 6769
 6770
 6771                                  ;;*****************************************************************
 6772                                  .SBTTL  PREPKB - PREPARE FOR KEYBOARD INPUT
 6773                                  ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
 6774                                  ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
 6775                                  ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
 6776                                  ;*ENABLES KBD INTERRUPT.
 6777                                  ;* CALL:
 6778                                  ;*      JSR     PC,PREPKB
```

```
                                    ;;*************************************************************
027302  005037  151640              PREPKB: CLR     @$TKB           ;CLEAR KBD BUFFER AND DONE BIT
027306  005037  005522                      CLR     INTCHR          ;CLEAR TTY INPUT WORD
027312  052737  000100  151624              BIS     #BIT6,@$TKS     ;ENABLE KBD INTERRUPT
027320  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
                                    ;;*************************************************************

027322  104401  005522              ECOBAD: TYPE    ,INTCHR         ;ECHO BAD CHARACTER
027326  104401  001314                      TYPE    ,$QUES          ;TYPE (?) AND <CR>, <LF>
027332  000207                              RTS     PC              ;RETURN



                                    ;;*************************************************************
                                    ;*SIZMEM - SIZE MEMORY, SET LIMITS
                                    ;*THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE, AND
                                    ;*IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
                                    ;*IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D).
                                    ;*AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70).
                                    ;;*************************************************************
027334  104407                      SIZMEM: SAVREG                  ;SAVE R0-R5
027336  012737  000200  056246              MOV     #200,$KT11      ;SET MEM MGT KEY FOR $SIZE
027344  004737  056210                      JSR     PC,$SIZE        ;SIZE MEMORY
027350  005737  056246                      TST     $KT11           ;SEE IF MEM MGT PRESENT
027354  100406                              BMI     8$              ;BR IF MEM MGT PRESENT
027356  013737  056512  005572              MOV     $LSTAD,MAHILM   ;SET MEM LIMIT LO BITS
027364  005037  005574                      CLR     MAHILM+2        ;CLEAR MEM LIMIT HI BITS
027370  000427                              BR      16$             ;GO TYPE LAST ADDRESS
                                    ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
027372  013700  056514              8$:     MOV     $LSTBK,R0       ;LO BITS
027376  005001                              CLR     R1              ;HI BITS
027400  012702  000006                      MOV     #6,R2           ;SET LOOP COUNT = 6
027404  000241                      12$:    CLC                     ;ROTATE LOOP
027406  006100                              ROL     R0
027410  006101                              ROL     R1
027412  005302                              DEC     R2
027414  001373                              BNE     12$
                                    ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
027416  063700  056512                      ADD     $LSTAD,R0       ;ADD LO BITS
027422  005501                              ADC     R1              ;HI BITS
027424  010037  005572                      MOV     R0,MAHILM       ;SET LO BITS OF MEM LIMIT
027430  010137  005574                      MOV     R1,MAHILM+2     ;SET HI BITS OF MEM LIMIT
027434  042701  000003                      BIC     #3,R1           ;CLEAR ADRS BITS 16,17
027440  001403                              BEQ     16$             ;BR IF NOT 22-BIT ADDRESSES
027442  112737  000001  003134              MOVB    #1,UBMPRS       ;SET "UNIBUS MAP PRESENT" FLAG
```

```
                                        ;TYPE "LAST PHYS MEM ADR = XXXXXXXX"
                        1$:     TYPE    .LSTMEM         ;TYPE "LAST PHYS MEM ADR ="
                                MOV     @MAHILM,-(SP)   ;PUT POINTER ON STACK
                                JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL ASCII
                                JSR     PC,@#$SUPRS     ;TYPE "XXXXXXXX"
                                TYPE    .$CRLF          ;TYPE <CR>,<LF>
                                TYPE    .$CRLF
                                RESREG                  ;RESTORE R0-R5
                                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.
                        ;;*******************************************************************
000002  016646  000002  GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
027510  104403                  TYPOS                   ;TYPE IT
027512     006                  .BYTE   6               ;SIX DIGITS
027513     000                  .BYTE   0               ;SUPPRESS LEADING ZEROS
027514  104401  007337          TYPE    ,NEWMSG         ;TYPE "  NEW = "
027520  004737  032272          JSR     PC,RDCHRS       ;READ NEW VALUE FROM KBD
027524  027556                  7$                      ;(!C) RETURN ADDRESS
027526  027556                  7$                      ;(!Z) RETURN ADDRESS
027530  027556                  7$                      ;(!U) OR ERROR RETURN ADDRESS
027532  005700                  TST     R0              ;SEE IF ANY CHARS TYPED
027534  001001                  BNE     4$              ;BR IF YES
027536  000207                  RTS     PC              ;RETURN - OLD VALUE UNCHANGED
027540  020027  000006  4$:     CMP     R0,#6           ;SEE IF > 6 CHARS TYPED
027544  003407                  BLE     8$              ;BR IF NOT BAD
027546  104401  005262  6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
027552  104401  001314          TYPE    ,$QUES
027556  162716  000010  7$:     SUB     #10,(SP)        ;FIX ERROR RETURN PC
027562  000207                  RTS     PC              ;ERROR RETURN
027564  012746  005262  8$:     MOV     #BUFFO,-(SP)    ;PUT POINTER TO CHARS ON STACK
027570  004737  053014          JSR     PC,OCTBIN       ;CONVERT DIGITS TO BINARY
027574  027546                  6$                      ;ERROR RETURN ADDRESS
027576  012600                  MOV     (SP)+,R0        ;GET NEW BINARY VALUE
027600  005737  053146          TST     $HIOCT          ;SEE IF HI BITS ARE 0
027604  001360                  BNE     6$              ;BR IF NOT
027606  010066  000002          MOV     R0,2(SP)        ;PUT NEW VALUE ON STACK
027612  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.
                        ;;*******************************************************************
027614  022737  000176  001140  GTSWRG: CMP     @#SWREG,SWR     ;SEE IF SOFTWARE SWR SELECTED
027622  001010                  BNE     6$              ;BR IF NOT
```

```
6891  027624  013746  000176                  MOV     SWREG,-(SP)     ;PUT OLD VALUE ON STACK
6892  027630  104401  007330                  TYPE    ,SWRMSG         ;TYPE "SWR = "
6893  027634  004737  027504                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
6894  027640  012637  000176                  MOV     (SP)+,SWREG     ;STORE NEW VALUE
6895  027644  000207              65:         RTS     PC              ;RETURN
6896
6897
6898
6899                                  ;;*****************************************************************
6900                                  ;*INITMM - INITIALIZE MEMORY MANAGEMENT REGISTERS
6901                                  ;*THIS SUBROUTINE INITIALIZES KIPAR'S AND KIPDR'S, FOR
6902                                  ;*CONTIGUOUS 4K PAGES, STARTING AT PHYSICAL ADDRESS 0.
6903                                  ;*KIPAR7 IS LOADED WITH 177600, TO PRESERVE THE I/O PAGE.
6904                                  ;*22-BIT MODE AND THE UNIBUS MAP ARE ENABLED, IF PRESENT,
6905                                  ;*BUT MEM MGT IS NOT TURNED ON IN THIS SUBROUTINE. HOWEVER,
6906                                  ;*MEM MGT TRAPS ARE ENABLED.
6907
6908                                  ;;*****************************************************************
6909  027646  104407              INITMM: SAVREG                  ;SAVE R0-R5
6910  027650  005001                      CLR     R1              ;INIT FOR PAR LOADING
6911  027652  012702  172340              MOV     #KIPAR0,R2      ;ADDR OF FIRST PAR
6912  027656  012703  000010              MOV     #8.,R3          ;LOAD 8 PAR'S AND 8 PDR'S
6913  027662  012762  077406  177740  4$: MOV     #77406,-40(R2)  ;PDR = 4K UP,READ/WRITE
6914  027670  010122                      MOV     R1,(R2)+        ;LOAD A PAR
6915  027672  062701  000200              ADD     #200,R1         ;UPDATE FOR NEXT PAR
6916  027676  005303                      DEC     R3              ;DECREMENT LOOP COUNTER
6917  027700  001370                      BNE     4$              ;LOOP UNTIL ALL 8 ARE LOADED
6918  027702  012742  177600              MOV     #177600,-(R2)   ;SET UP KIPAR7 FOR I/O PAGE
6919  027706  105737  003134              TSTB    UBMPRS          ;SEE IF 22-BIT ADDRESSES
6920  027712  001403                      BEQ     10$             ;BR IF NOT
6921  027714  012737  000060  172516      MOV     #60,@#SR3       ;ENABLE 22-BIT MODE,AND UNIBUS MAP
6922  027722  012737  001000  177572  10$: MOV     #BIT9,@#SR0     ;ENABLE KT11 TRAPS
6923  027730  104410                      RESREG                  ;RESTORE R0-R5
6924  027732  000207                      RTS     PC              ;RETURN
6925
6926
6927
6928                                  ;;*****************************************************************
6929                                  ;*SERVICE ROUTINE FOR MEM MGT TRAPS
6930                                  ;;*****************************************************************
6931  027734  004737  042206          KTERHD: JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
6932  027740  013737  177572  001174          MOV     @#SR0,$REG5     ;GET SR0
6933  027746  013737  177574  001176          MOV     @#SR1,$REG6     ;GET SR1
6934  027754  013737  177576  001200          MOV     @#SR2,$REG7     ;GET SR2
6935  027762  012737  030006  000004          MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
6936  027770  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
6937  027776  013737  172516  001202          MOV     @#SR3,$REG10    ;GET SR3
6938  030004  000406                      BR      10$
6939  030006  022626              8$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
6940  030010  112737  000003  064612      MOVB    #3,@#DF30+18.   ;FIX PRINTOUT FOR NO SR3
6941  030016  105037  063253              CLRB    @#DH70+11.
6942  030022  104121              10$:    ERROR   121             ;KT11 FAILURE
6943  030024  000137  044476              JMP     HLTPRG          ;ABORT !!!
6944
6945
6946
```

```
                                        ;;*****************************************************************
                                        ;*ENBCSR - ENABLE MEMORY CSR'S FOR PARITY ERRORS
                                        ;;*****************************************************************
030030  010146                  ENBCSR: MOV     R1,-(SP)        ;SAVE R1
030032  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
030036  013746  000006                  MOV     @#ERRVEC+2,-(SP)
030042  012737  030064  000004          MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
030050  012701  172100                  MOV     #MEMCSR,R1      ;ADRS OF MEMORY CSR'S
030054  005011                  4$:     CLR     (R1)            ;ZERO THIS CSR
030056  012711  000001                  MOV     #BIT0,(R1)      ;SET ENABLE IN THIS CSR
030062  000401                          BR      10$
030064  022626                  8$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
030066  062701  000002          10$:    ADD     #2,R1           ;INCREMENT TO NEXT CSR ADRS
030072  020127  172140                  CMP     R1,#MEMCSR+40   ;SEE IF DONE CHECKING YET
030076  001366                          BNE     4$              ;BR IF NOT
030100  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;RESTORE OLD VECTORS
030104  012637  000004                  MOV     (SP)+,@#ERRVEC
030110  012601                          MOV     (SP)+,R1        ;RESTORE R1
030112  000207                          RTS     PC              ;RETURN

                                        ;;*****************************************************************
                                        ;*SERVICE ROUTINE FOR MEM PARITY ERRORS
                                        ;;*****************************************************************
030114  010146                  MPERHD: MOV     R1,-(SP)        ;SAVE R1
030116  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
030122  013746  000006                  MOV     @#ERRVEC+2,-(SP)
030126  004737  042206                  JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
030132  016637  000006  001174          MOV     6(SP),$REG5     ;GET PC OF ERROR
030140  012737  030264  000004          MOV     #10$,@#ERRVEC   ;SET T.O. VECTOR
030146  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
                                ;HANDLE 11/70 MEMORY PARITY ERROR
030154  013737  177740  001176          MOV     @#LOERAD,$REG6  ;LOW ERROR ADRS REG
030162  013737  177742  001200          MOV     @#HIERAD,$REG7  ;HI ERROR ADRS REG
030170  013737  177744  001202          MOV     @#MEMSYS,$REG10 ;MEMORY SYSTEM REG
030176  012737  063307  064634          MOV     #DH707,DF31+16. ;FIX ERROR MSG FOR 11/70
030204  112737  000004  064636          MOVB    #4,DF31+18.
030212  104122                          ERROR   122             ;11/70 MEM PARITY ERROR
030214  005737  001200                  TST     $REG7           ;SEE IF BAD MEMORY IS IN PROGRAM AREA
030220  001011                          BNE     6$              ;BR IF NOT
030222  023727  001176  072640          CMP     $REG6,#RWBUF+E000 ;SEE IF BAD MEM IS IN PROG. AREA
030230  103005                          BHIS    6$              ;BR IF NOT
030232  105737  003135          4$:     TSTB    MEMABT          ;SEE IF ABORT DESIRED
030236  001002                          BNE     6$              ;BR IF NOT
030240  000137  044476                  JMP     HLTPRG          ;ABORT !!!
030244  012637  000006          6$:     MOV     (SP)+,@#ERRVEC+2 ;RESTORE T.O. VECTOR
030250  012637  000004                  MOV     (SP)+,@#ERRVEC
030254  012601                          MOV     (SP)+,R1        ;RESTORE R1
030256  004737  030030                  JSR     PC,ENBCSR       ;GO CLEAR AND ENABLE CSR'S
030262  000002                          RTI                     ;RETURN
                                ;HANDLE ALL OTHER MEMORY PARITY ERRORS  (NON-11/70)
030264  022626                  10$:    CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
030266  012737  030316  000004          MOV     #14$,@#ERRVEC   ;SET T.O. VECTOR
030274  012701  172100                  MOV     #MEMCSR,R1      ;GET FIRST CSR ADDRESS
030300  011137  001200          12$:    MOV     (R1),$REG7      ;CHECK FOR A MEMORY CSR
```

```
030304  100005                          BPL     16$             ;BR IF NO ERROR SET HERE
030306  013737  001176                  MOV     R1,$REG6        ;GET CSR ADDRESS
030312  104122                          ERROR   122             ;MEMORY PARITY ERROR (NON-11/70)
030314  000746                          BR      4$              ;GO SEE IF SHOULD ABORT
030316  022626                  14$:    CMP     (SP)+,(SP)+     ;CLEAN UP STACK
030320  062701  000002            16$:    ADD     #2,R1           ;INCR R1 TO POINT TO NEXT CSR
030324  020127  172140                    CMP     R1,#MEMCSR+40   ;SEE IF DONE CHECKING
030330  103763                          BLO     12$             ;BR IF NOT
030332  000744                          BR      6$              ;RETURN


                                ;;*****************************************************************
                                ;*PREPAR - PREPARE MEM MGT FOR RELOCATION
                                ;*THIS SUBROUTINE CALLS INITMM, AND THEN SETS UP CONSTANT
                                ;*FOR KIPAR6 IN SAVPAR, AND LOADS THE UNIBUS MAP REGISTERS,
                                ;*IF PRESENT.
                                ;;*****************************************************************
030334  104407                  PREPAR: SAVREG                  ;SAVE R0-R5
030336  004737  027646                  JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
030342  013700  005602                  MOV     PMA,R0          ;LO BITS OF MA
030346  042700  017777                  BIC     #17777,R0       ;MASK FOR BITS 13-15
030352  013701  005604                  MOV     PMA+2,R1        ;HI BITS OF MA
030356  010003                          MOV     R0,R3           ;SAVE THESE BITS
030360  010104                          MOV     R1,R4
030362  006100                          ROL     R0              ;GET MA BITS 13-21 INTO R1 BITS 7-15
030364  006101                          ROL     R1
030366  006100                          ROL     R0
030370  006101                          ROL     R1
030372  000301                          SWAB    R1
030374  006100                          ROL     R0
030376  106001                          RORB    R1
030400  010137  003176                  MOV     R1,SAVPAR       ;CONSTANT FOR LOADING PAR6 LATER
030404  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP ENABLED
030410  001420                          BEQ     9$              ;BR IF NOT (NO UNIBUS MAPPING)
                                ;PREPARE FOR 22-BIT ADDRESSING -  LOAD UNIBUS MAP REGISTERS
030412  012700  170200                  MOV     #MAPLOW,R0      ;STARTING ADDR OF MAP REGISTERS
030416  012701  000037                  MOV     #31.,R1         ;SET REGISTER COUNTER
030422  010320                  4$:     MOV     R3,(R0)+        ;LOAD A MAP REGISTER
030424  010420                          MOV     R4,(R0)+
030426  062703  020000                  ADD     #20000,R3       ;ADD 4K WORDS
030432  005504                          ADC     R4
030434  077106                          SOB     R1,4$           ;LOOP UNTIL 31(DEC) ARE LOADED
030436  042765  160000  000010          BIC     #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
030444  142765  000003  000007          BICB    #3,P.BAHI(R5)
030452  104410                  9$:     RESREG                  ;RESTORE R0-R5
030454  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;* FNDXDP - FIND STARTING ADR OF XXDP LOADER , AND STORE IT
                                ;*IN XXDPAD.
                                ;;*****************************************************************
030456  005737  005574          FNDXDP: TST     MAHILM+2        ;TEST HI BITS OF UPPER MEMORY LIMIT
030462  001404                          BEQ     6$              ;BR IF HI BITS ARE 0
```

```
 7059  030464  012737  160000  005536  4$:     MOV     #160000,XXDPAD  ;START OF 28K IS END OF XXDP
 7060  030472  000412                          BR      8$
 7061  030474  023727  005572  157776  6$:     CMP     MAHILM,#157776  ;SEE IF MEM LIMIT > OR = 157776
 7062  030502  103370                          BHIS    4$              ;BR IF YES
 7063  030504  013737  005572  005536          MOV     MAHILM,XXDPAD   ;HI MEM LIMIT IS END OF XXDP
 7064  030512  062737  000002  005536          ADD     #2,XXDPAD       ;ADD 2 BYTES
 7065  030520  162737  006000  005536  8$:     SUB     #6000,XXDPAD    ;COMPUTE START OF XXDP
 7066  030526  000207                          RTS     PC              ;RETURN
 7067
 7068
 7069
 7070                                          ;;*****************************************************************
 7071                                          ;* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
 7072                                          ;*THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
 7073                                          ;*PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
 7074                                          ;*STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
 7075                                          ;*
 7076                                          ;* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
 7077                                          ;*THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
 7078                                          ;*STORED IN XDPSAV,XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
 7079                                          ;*ADDRESS).
 7080                                          ;;*****************************************************************
 7081  030530  104407                  SAVXDP: SAVREG                  ;SAVE R0-R5
 7082  030532  005004                          CLR     R4              ;SET INDICATOR TO SAVE XXDP
 7083  030534  000410                          BR      XDP1
 7084  030536  104407                  GETXDP: SAVREG                  ;SAVE R0-R5
 7085  030540  105037  003124                  CLRB    XOVLAD          ;CLEAR XXDP OVERLAID INDICATOR
 7086  030544  105737  003125                  TSTB    XDPSVD          ;SEE IF XXDP WAS SAVED
 7087  030550  001425                          BEQ     XDP2            ;BR IF NOT SAVED
 7088  030552  012704  000001                  MOV     #1,R4           ;SET INDICATOR TO GET XXDP
 7089  030556  005737  000170          XDP1:   TST     KILLDR          ;SEE IF OK TO OVERLAY LOADER
 7090  030562  001020                          BNE     XDP2            ;BR IF YES, TO EXIT
 7091  030564  012702  003000                  MOV     #1536.,R2       ;GET SET TO MOVE 1536(DEC) WORDS
 7092  030570  013703  005536                  MOV     XXDPAD,R3       ;GET ORIG ADR OF START OF XXDP
 7093  030574  013700  005540                  MOV     XDPSAV,R0       ;GET SAVE ADR LO BITS
 7094  030600  013701  005542                  MOV     XDPSAV+2,R1     ;HI BITS
 7095  030604  005737  056246                  TST     $KTI1           ;SEE IF MEM MGT PRESENT
 7096  030610  100417                          BMI     XDP3            ;BR IF PRESENT
 7097  030612  005704                          TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
 7098  030614  001005                          BNE     XDP4            ;BR IF WANT TO GET XXDP
 7099  030616  012320                  6$:     MOV     (R3)+,(R0)+     ;SAVE A WORD
 7100  030620  005302                          DEC     R2              ;SEE IF 1536(DEC) WORDS YET
 7101  030622  001375                          BNE     6$              ;BR IF NOT YET
 7102  030624  104410                  XDP2:   RESREG                  ;RESTORE R0-R5
 7103  030626  000207                          RTS     PC              ;RETURN
 7104  030630  062700  006000          XDP4:   ADD     #6000,R0        ;POINT TO END OF SAVE AREA
 7105  030634  062703  006000                  ADD     #6000,R3        ;POINT TO END OF XXDP LOADER AREA
 7106  030640  014043                  8$:     MOV     -(R0),-(R3)     ;GET A WORD
 7107  030642  005302                          DEC     R2              ;SEE IF 1536(DEC) WORDS YET
 7108  030644  001375                          BNE     8$              ;BR IF NOT YET
 7109  030646  000766                          BR      XDP2            ;GO EXIT
 7110                                  ;COME HERE IF MEM MGT
 7111  030650  004737  027646          XDP3:   JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
 7112  030654  006100                          ROL     R0              ;GET ADR BITS 13-21 INTO R1 BITS 7-15
 7113  030656  006101                          ROL     R1
 7114  030660  006100                          ROL     R0
```

```
 7115  030662  006101                        ROL     R1
 7116  030664  000301                        SWAB    R1
 7117  030666  006100                        ROL     R0
 7118  030670  106001                        RORB    R1
 7119  030672  010137  172354                MOV     R1,@#KIPAR6     ;SET UP PAR6
 7120  030676  013701  005540                MOV     XDFSAV,R1       ;GET LO ADRS BITS
 7121  030702  042701  160000                BIC     #160000,R1      ;FIX UP R1 TO REFERENCE PAR6
 7122  030706  052701  140000                BIS     #140000,R1
 7123  030712  005704                16$:    TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
 7124  030714  001007                        BNE     18$             ;BR IF WANT TO GET XXDP
 7125  030716  012305                        MOV     (R3)+,R5        ;SAVE A WORD
 7126  030720  005237  177572                INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
 7127  030724  010521                        MOV     R5,(R1)+
 7128  030726  005337  177572                DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
 7129  030732  000406                        BR      20$
 7130  030734  005237  177572          18$:  INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
 7131  030740  012105                        MOV     (R1)+,R5        ;GET A WORD
 7132  030742  005337  177572                DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
 7133  030746  010523                        MOV     R5,(R3)+
 7134  030750  032701  020000          20$:  BIT     #BIT13,R1       ;SEE IF OVERFLOW TO PAGE 7
 7135  030754  001405                        BEQ     22$             ;BR IF NOT
 7136  030756  042701  020000                BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 7137  030762  062737  000200  172354        ADD     #200,@#KIPAR6   ;UPDATE PAR BY 4K
 7138  030770  005302                22$:    DEC     R2              ;SEE IF 1536 WORDS YET
 7139  030772  001347                        BNE     16$             ;BR IF NOT YET
 7140  030774  000713                        BR      XDP2            ;BR TO RETURN
 7141
 7142
 7143
 7144                                ;;*****************************************************************
 7145                                .SBTTL  INITSS - INITIALIZE SUBSYSTEM
 7146                                ;*
 7147                                ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
 7148                                ;*AND DOES A SUBSYSTEM CLEAR.
 7149                                ;* USES -   R2,R5
 7150                                ;* CALL:
 7151                                ;*              JSR     PC,INITSS
 7152                                ;;*****************************************************************
 7153
 7154  030776  012737  042466  003036  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
 7155  031004  012737  041220  003034          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
 7156  031012  013702  003026                 MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
 7157  031016  012705  002620                 MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
 7158  031022  105037  003133                 CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
 7159  031026  004737  031070                 JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
 7160  031032  112765  000177  000001         MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
 7161  031040  004737  040774                 JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
 7162  031044  113765  005500  000000         MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
 7163  031052  113737  005500  002704         MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
 7164  031060  113765  003115  000007         MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
 7165  031066  000207                         RTS     PC              ;SUBROUTINE EXIT
 7166
 7167
 7168
 7169                                ;;*****************************************************************
 7170                                .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
                                ;;*****************************************************************

031070  010046                  CLRPRM: MOV     R0,-(SP)        ;SAVE R0
031072  010546                          MOV     R5,-(SP)        ;SAVE R5
031074  010500                          MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
031076  062705  000016                  ADD     #P.CSI,R5       ;COMPUTE LIMIT ADDRESS
031102  005020                  1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
031104  020005                          CMP     R0,R5           ;SEE IF DONE YET
031106  001375                          BNE     1$              ;BR IF  NOT DONE  YET
031110  012605                          MOV     (SP)+,R5        ;RESTORE R5
031112  012600                          MOV     (SP)+,R0        ;RESTORE R0
031114  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  SCNDRV - SCAN DRIVE FOR STATUS
                                ;*THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
                                ;*THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
                                ;*AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
                                ;*OF THESE CONDITIONS ARE NOT MET, AN APPROPRIATE
                                ;*MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
                                ;*A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
                                ;*AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
                                ;*MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
                                ;*ALSO, IF RUNNING FROM ADRS 200 AND RK06 IS LOAD MEDIUM,
                                ;*DRIVE 0 WILL BE REJECTED FOR USE.
                                ;* CALL -    JSR     PC,SCNDRV
                                ;*          <ERROR RETURN ADDRESS>
                                ;*
                                ;;*****************************************************************

031116  004737  030776          SCNDRV: JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
031122  113765  000500  000000          MOVB    DRIVE,P.DRVN(R5) ;GET DRIVE NUMBER
                                ;SEE IF DRIVE 0 IS XXDP LOAD MEDIUM
031130  001012                          BNE     3$              ;BR IF NOT DRIVE 0
031132  122737  000013  000041          CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
031140  001006                          BNE     3$              ;BR IF NOT
031142  105737  003106                  TSTB    MDFLAG          ;SEE IF 200 START
031146  001003                          BNE     3$              ;BR IF NOT
031150  104401  007673                  TYPE    ,DROXDP         ;TYPE "DRIVE 0 IS LOAD MEDIUM"
031154  000434                          BR      2$              ;TAKE ERROR EXIT
031156  112765  000141  000001  3$:     MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
031164  012737  027064  003036          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
031172  012737  000377  005530          MOV     #377,NEWON      ;INIT. ON-LINE INDICATOR
031200  004737  040774                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
                                ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
031204  012737  042466  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNL RETURN ADDRESS
031212  022737  000377  005530          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
031220  001420                          BEQ     4$              ;BR IF NED NOT SET
031222  113737  005500  007374          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
031230  152737  000060  007374          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
031236  104401  007366                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
```

```
7227  031242  104401  007377                  TYPE    ,NXDRIV          ;TYPE "NON-EXISTENT"
7228                                ;SERVICE ERRORS HERE
7229  031246  042762  000100  000000  2$:     BIC     #IE,RKCS1(R2)    ;DISABLE RK06 INTERRUPT
7230  031254  017616  000000            2$:   MOV     @(SP),.SP)       ;SET UP ERROR RETURN ADDRESS
7231  031260  000207                          RTS     PC               ;ERROR RETURN
7232                                ;SEE IF DRIVE IS READY
7233  031262  032765  000200  000040  4$:     BIT     #S.DRY,P.ADD(R5) ;TEST FOR DRIVE READY
7234  031270  001013                          BNE     6$               ;BR IF DRIVE IS READY
7235  031272  113737  005500  007374          MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO. INTO BUF
7236  031300  152737  000060  007374          BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
7237  031306  104401  007366                  TYPE    ,BADDRV          ;TYPE "DRIVE X"
7238  031312  104401  007416                  TYPE    ,NTREDY          ;TYPE "NOT READY"
7239  031316  000753                          BR      2$               ;TAKE ERROR EXIT
7240                                ;SEE IF DRIVE IS WRITE ENABLED
7241  031320  032765  004000  000040  6$:     BIT     #S.WRL,P.ADD(R5) ;SEE IF WRITE LOCK SET
7242  031326  001413                          BEQ     8$               ;BR IF WRITE LOCK NOT SET
7243  031330  113737  005500  007374          MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO.
7244  031336  152737  000060  007374          BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
7245  031344  104401  007366                  TYPE    ,BADDRV          ;TYPE "DRIVE X"
7246  031350  104401  007432                  TYPE    ,WRTLOK          ;TYPE "WRITE-LOCKED"
7247  031354  000734                          BR      2$               ;TAKE ERROR EXIT
7248                                ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
7249  031356  112765  000103  000001  8$:     MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7250  031364  004737  040774                  JSR     PC,DRVCAL        ;SET VOLUME VALID
7251  031370  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
7252  031376  012765  000632  000002          MOV     #LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)
7253  031404  112765  000002  000005          MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
7254  031412  012765  064640  000010          MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
7255  031420  012765  177774  000012          MOV     #-4,P.WC(R5)     ;READ 4 WORDS
7256  031426  105065  000007                  CLRB    P.CS1H(R5)       ;SET 22-SECTOR FORMAT
7257  031432  004737  040774                  JSR     PC,DRVCAL        ;READ 4 WORDS OF BAD SECTOR FILE
7258  031436  032737  100000  005474          BIT     #ANYDER,RECODE   ;SEE IF DATA ERROR
7259  031444  001402                          BEQ     10$              ;BR IF OK
7260  031446  104401  013314                  TYPE    ,BAD632          ;TYPE READ ERROR MESSAGE
7261  031452  022737  177777  064646  10$:     CMP     #177777,RWBUF+6  ;SEE IF ALL 1'S IN I.D. WORD 3
7262  031460  001013                          BNE     12$              ;BR IF NOT ALL 1'S
7263  031462  113737  005500  007374          MOVB    DRIVE,BADDRV+6   ;GET DRIVE NO.
7264  031470  152737  000060  007374          BISB    #'0,BADDRV+6     ;CONVERT TO ASCII
7265  031476  104401  007366                  TYPE    ,BADDRV          ;TYPE "DRIVE X"
7266  031502  104401  007451                  TYPE    ,ALNPAK          ;TYPE "LOADED WITH ALIGN PACK"
7267  031506  000657                          BR      2$               ;TAKE ERROR EXIT
7268                                ;ERROR FREE RETURN
7269  031510  112765  000113  000001  12$:    MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
7270  031516  004737  040774                  JSR     PC,DRVCAL        ;RECALIBRATE THIS DRIVE
7271  031522  062716  000002                  ADD     #2,(SP)          ;FIX UP RETURN PC
7272  031526  000207                          RTS     PC               ;RETURN
7273
7274
7275
7276                                ;;*****************************************************************
7277                                .SBTTL  CHKDRV - CHECK STATUS OF DRIVE
7278                                ;*THIS SUBROUTINE CHECKS THE DESIRED DRIVE TO DETERMINE
7279                                ;*IF THE DRIVE IS ON-LINE,READY,WRITE-PROTECTED. IF NOT, THE
7280                                ;*APPROPPRIATE ERROR MESSAGE IS TYPED,AND THE CPU HALTS
7281                                ;*WHILE MANUAL  INTERVENTION TAKES PLACE TO CORRECT THE
7282                                ;*PROBLEM. THE OPERATOR THEN DEPRESSES 'CONT' TO  PROCEED.
```

```
                                        ;*AND A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
                                        ;*AFTER THE CALL. IF THERE ARE NO  PROBLEMS WITH THIS
                                        ;*DRIVE, A NORMAL  RETURN IS MADE.
                                        ;*R5 MUST CONTAIN THE ADDRESS OF THE PARAMETER BLOCK,
                                        ;*AND R2 MUST CONTAIN THE RK06 REGISTER BASE ADDRESS.
                                        ;*
                                        ;* CALL:
                                        ;*      JSR     PC,CHKDRV
                                        ;*      ERROR ADDRESS
                                        ;;*****************************************************************
031530  113765  005500  000000  CHKDRV: MOVB    DRIVE,P.DRVN(R5) ;SET DESIRED DRIVE NUMBER
031536  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
031544  012737  027064  003036          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
031552  012737  000377  005530          MOV     #377,NEWON      ;INITIALIZE ON-LINE INDICATOR
031560  004737  040774                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
                                        ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
031564  012737  042466  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
031572  022737  000377  005530          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
031600  001411                          BEQ     1$              ;BR IF NED NOT SET
031602  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
031606  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
031612  110137  012043                  MOVB    R1,DRVNED       ;PUT DRIVE NO. INTO MSG BUFFER
031616  104401  012031                  TYPE    ,NONEXD         ;TYPE NON-EXISTENT DRIVE MESSAGE
031622  000414                          BR      3$              ;SERVICE THE ERROR
                                        ;SEE IF DESIRED DRIVE IS READY
031624  032765  000200  000040  1$:     BIT     #S.DRY,P.ADD(R5) ;TEST FOR DRIVE READY
031632  001024                          BNE     4$              ;BR IF DRIVE READY
031634  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
031640  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
031644  110137  011677                  MOVB    R1,DRNRDY       ;PUT DRIVE NUMBER IN MESSAGE BUFFER
031650  104401  011665                  TYPE    ,NOTRDY         ;TYPE DRIVE NOT READY MESSAGE
                                        ;SERVICE ALL ERRORS HERE
031654  042762  000100  000000  3$:     BIC     #IE,RKCS1(R2)   ;CLEAR RK06 INTERRUPT ENABLE BIT
031662  000000                          HALT                    ;HALT IS NECESSARY DURING MANUAL
                                                                ;  INTERVENTIONS, TO CHANGE PACK,
                                                                ;  WRITE LOCK,START DRIVE,ETC.
                                                                ;PRESS 'CONT' TO PROCEED !
031664  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
031672  004737  040774                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
031676  017616  000000                  MOV     @(SP),(SP)      ;SET UP ERROR RETURN ADDRESS
031702  000207                          RTS     PC              ;ERROR RETURN
                                        ;SEE IF DRIVE IS WRITE-LOCKED
031704  032765  004000  000040  4$:     BIT     #S.WRL,P.ADD(R5) ;SEE IF WRITE LOCK SET
031712  001011                          BNE     5$              ;BR IF WRITE-PROTECTED
031714  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
031720  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
031724  110137  012172                  MOVB    R1,NODRLK       ;PUT DRIVE NUMBER IN MSG BUFFER
031730  104401  012160                  TYPE    ,NOTLOK         ;TYPE "DRIVE NOT WRITE-LOCKED"
031734  000747                          BR      3$              ;SERVICE THE ERROR
                                        ;RECALIBRATE DESIRED DRIVE, SET VOLUME VALID
031736  112765  000113  000001  5$:     MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
031744  004737  040774                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
031750  112765  000103  000001          MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
031756  004737  040774                  JSR     PC,DRVCAL       ;DO PACK ACK. (SETS VOLUME VALID)
                                        ;RETURN HERE
```

```
 7339  031762  062716  000002          7$:     ADD     #2,(SP)         ;FIX UP RETURN ADDRESS ON STACK
 7340  031766  000207                          RTS     PC              ;ERROR-FREE RETURN
 7341
 7342
 7343
 7344                                  ;;*****************************************************************
 7345                                  ;* ALNSEK - SEEK IN SINGLE INCREMENTS FROM CYL 0 TO ALIGNMENT CYL (365 OCT).
 7346                                  ;;*****************************************************************
 7347  031770  112765  000117  000001  ALNSEK: MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 7348  031776  005065  000002                  CLR     P.CYLN(R5)      ;INIT CYL TO 0
 7349  032002  004737  040774          2$:     JSR     PC,DRVCAL       ;SEEK TO THIS CYL
 7350  032006  005265  000002                  INC     P.CYLN(R5)      ;INCR CYL NO.
 7351  032012  022765  000366  000002          CMP     #ALNCYL+1,P.CYLN(R5) ;SEE IF 365 REACHED YET
 7352  032020  001370                          BNE     2$              ;BR IF NOT YET
 7353  032022  000207                          RTS     PC              ;RETURN
 7354
 7355
 7356
 7357                                  ;;*****************************************************************
 7358                                  ;* WAITHR - THIS SUBROUTINE UNLOADS HEADS ON THE CURRENT DRIVE, TYPES
 7359                                  ;* "TYPE <R> WHEN READY", AND THEN LOADS THE HEADS WHEN <R> IS TYPED.
 7360                                  ;;*****************************************************************
 7361  032024  112765  000107  000001  WAITHR: MOVB    #UNLOAD,P.CMND(R5) ;SET UNLOAD COMMAND
 7362  032032  004737  040774                  JSR     PC,DRVCAL       ;UNLOAD HEADS ON THIS DRIVE
 7363  032036  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
 7364  032044  004737  040774          1$:     JSR     PC,DRVCAL       ;READ STATUS OF DRIVE
 7365  032050  032765  000040  000044          BIT     #S.HDHM,P.AD1(R5) ;SEE IF HEADS UNLOADED YET
 7366  032056  001772                          BEQ     1$              ;BR IF NOT YET
 7367  032060  104401  012755          2$:     TYPE    ,RWNRDY         ;TYPE "TYPE <R> WHEN READY"
 7368  032064  004737  027302                  JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 7369  032070  005737  005522          4$:     TST     INTCHR          ;SEE IF ANY INPUT YET.
 7370  032074  001775                          BEQ     4$              ;BR IF NOT YET
 7371  032076  022737  000122  005522          CMP     #'R,INTCHR      ;SEE IF <R> TYPED
 7372  032104  001403                          BEQ     6$              ;BR IF <R> TYPED
 7373  032106  004737  027322                  JSR     PC,ECOBAD       ;ECHO BAD INPUT
 7374  032112  000762                          BR      2$              ;GO ASK AGAIN
 7375  032114  112765  000111  000001  6$:     MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
 7376  032122  004737  040774                  JSR     PC,DRVCAL       ;START SPINDLE AND LOAD HEADS
 7377  032126  000207                          RTS     PC              ;RETURN
 7378
 7379
 7380
 7381                                  ;;*****************************************************************
 7382                                  ;*SETTUP - SET UP FOR LOOP ON ERROR
 7383                                  ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
 7384                                  ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
 7385                                  ;;*****************************************************************
 7386  032130  011637  001076          SETTUP: MOV     (SP),@#STACK-2  ;MOVE RETURN PC ON STACK
 7387  032134  012706  001076                  MOV     #STACK-2,SP     ;RE-INIT THE STACK POINTER
 7388  032140  005037  005474                  CLR     RECODE          ;CLEAR ERROR RECOVERY FLAGS
 7389  032144  105037  003116                  CLRB    ERRCNT          ;CLEAR RETRY ERROR COUNT
 7390  032150  012705  002620                  MOV     #PARM0,R5       ;SET PARAM BLK ADRS
 7391  032154  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
 7392  032162  004737  040774                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
 7393  032166  012737  000000  177776          MOV     #PR0,@#PS       ;RE-ESTABLISH PRIORITY 0
 7394  032174  000207                          RTS     PC              ;RETURN
```

```
                        ;;*****************************************************************
                        .SBTTL  CHKITR - CHECK CURRENT TEST ITERATION NUMBER
                        ;*THIS SUBROUTINE LOADS THE ITERATION NUMBER FOR THE CURRENT
                        ;*TEST INTO $TIMES, AND CHECKS IT TO DETERMINE IF THIS TEST
                        ;*SHOULD BE RUN. IF ITERATION NO. = 0, THE SUBROUTINE RETURNS
                        ;*TO THE STORED PC (AND THE TEST WILL BE SKIPPED). IF IT IS
                        ;*NOT 0, THE RETURN IS MADE TO PC+2, AND THE TEST WILL
                        ;*BE RUN. HOWEVER, ON THE FIRST PASS (QUICK VERIFY) OF A RUN
                        ;*WHICH STARTED AT ADRS 200, ALL TESTS WILL BE RUN ONCE.
                        
032176  010146          CHKITR: MOV     R1,-(SP)        ;SAVE R1
032200  112737  000144  001115        MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
032206  105737  003126              TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
032212  001006                      BNE     3$              ;BR IF YES
032214  105737  003106              TSTB    MDFLAG          ;SEE IF 200 START
032220  001007                      BNE     4$              ;BR IF NOT
032222  005737  001326              TST     $PASS           ;SEE IF FIRST PASS
032226  001004                      BNE     4$              ;BR IF NOT
032230  012737  000001  001304  3$:     MOV     #1,$TIMES       ;SET UP FOR 1 ITERATION
032236  000410                      BR      1$              ;GO RUN DUAL-ACCESS TEST
032240  113701  001102          4$:     MOVB    $TSTNM,R1       ;GET CURRENT TEST NUMBER
032244  005301                      DEC     R1              ;DECREMENT BY 1
032246  006301                      ASL     R1              ;DOUBLE IT, TO GET TEST LIST INDEX
032250  016137  005616  001304      MOV     TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
032256  001403                      BEQ     2$              ;BR IF TEST SHOULD BE SKIPPED
032260  062766  000004  000002  1$:     ADD     #4,2(SP)        ;ADJUST RETURN PC TO RUN THIS TEST
032266  012601                  2$:     MOV     (SP)+,R1        ;RESTORE R1
032270  000207                      RTS     PC              ;RETURN
                        
                        
                        
                        ;;*****************************************************************
                        .SBTTL  RDCHRS - READ A STRING OF KBD INPUT CHARS
                        ;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
                        ;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
                        ;*THEM IN BUFFO, TERMINATED BY A NULL (0) BYTE.
                        ;*RUB-OUT AND (↑U) FEATURES ARE PROVIDED.
                        ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
                        ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
                        ;*TAKEN IF (↑C) IS TYPED, THE SECOND IS TAKEN IF (↑Z) IS
                        ;*TYPED, AND THE THIRD IS TAKEN IF (↑U) OR INVALID INPUT IS TYPED.
                        ;*IF (↑G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
                        ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (↑G) RETURN
                        ;*IS TAKEN.
                        ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
                        ;*IN R0.
                        ;*
                        ;*  CALL -      JSR     PC,RDCHRS
                        ;*              <CONTROL-C RETURN ADDRESS>
                        ;*              <CONTROL-Z RETURN ADDRESS>
                        ;*              <CONTROL-U OR ERROR RETURN ADDRESS>
                        ;*              RETURN
                        ;;*****************************************************************
```

```
7451
7452  032272                          RDCHRS:
7453  032272  010146                          MOV     R1,-(SP)        ;SAVE R1
7454  032274  010246                          MOV     R2,-(SP)        ;SAVE R2
7455  032276  005000                          CLR     R0              ;INITIALIZE CHARACTER COUNT
7456  032300  005001                          CLR     R1              ;INITIALIZE RUB-OUT INDICATOR
7457                                  ;READ A CHARACTER
7458  032302  104406                  2$:     RDCHR                   ;READ A CHARACTER
7459  032304  112602                          MOVB    (SP)+,R2        ;GET CHARACTER INTO R2
7460                                  ;CHECK FOR (↑C)
7461  032306  122702  000003                  CMPB    #003,R2         ;SEE IF (↑C) TYPED
7462  032312  001006                          BNE     4$              ;BR IF NOT (↑C)
7463  032314  104401  013360                  TYPE    ,CNTRLC         ;ECHO (↑C)
7464  032320  017666  000004  000004  3$:     MOV     @4(SP),4(SP)    ;PUT RETURN ADDRESS ON STACK
7465  032326  000523                          BR      24$             ;BR TO TAKE EXIT
7466                                  ;CHECK FOR (↑Z)
7467  032330  122702  000032          4$:     CMPB    #032,R2         ;SEE IF (↑Z) TYPED
7468  032334  001006                          BNE     6$              ;BR IF NOT (↑Z)
7469  032336  104401  013365                  TYPE    ,CNTRLZ         ;ECHO (↑Z)
7470  032342  062766  000002  000004          ADD     #2,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADR.
7471  032350  000763                          BR      3$              ;BR TO TAKE (↑Z) EXIT
7472                                  ;CHECK FOR (↑U)
7473  032352  122702  000025          6$:     CMPB    #025,R2         ;SEE IF (↑U) TYPED
7474  032356  001006                          BNE     8$              ;BR IF NOT (↑U)
7475  032360  104401  013377                  TYPE    ,CNTRLU         ;ECHO (↑U)
7476  032364  062766  000004  000004  7$:     ADD     #4,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
7477  032372  000752                          BR      3$              ;BR TO TAKE (↑U) EXIT
7478                                  ;CHECK FOR (↑G)
7479  032374  122702  000007          8$:     CMPB    #007,R2         ;SEE IF (↑G) TYPED
7480  032400  001005                          BNE     9$              ;BR IF NOT (↑G)
7481  032402  104401  013404                  TYPE    ,CNTRLG         ;ECHO (↑G)
7482  032406  004737  027614                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
7483  032412  000764                          BR      7$              ;TAKE (↑U) RETURN
7484                                  ;CHECK FOR RUB-OUT (DELETE)
7485  032414  122702  000177          9$:     CMPB    #177,R2         ;SEE IF RUB-OUT (DEL) TYPED
7486  032420  001020                          BNE     14$             ;BR IF NOT RUB-OUT
7487  032422  005700                          TST     R0              ;CHECK THE CHARACTER COUNT
7488  032424  001726                          BEQ     2$              ;BR IF COUNT = 0
7489  032426  005701                          TST     R1              ;CHECK THE RUB-OUT INDICATOR
7490  032430  001003                          BNE     11$             ;BR IF WE HAD A PREVIOUS RUB-OUT
7491  032432  005201                          INC     R1              ;SET RUB-OUT INDICATOR
7492  032434  104401  013415                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH (\)
7493  032440  005037  005532          11$:    CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
7494  032444  005300                          DEC     R0              ;DECREMENT COUNT
7495  032446  116037  005262  005532          MOVB    BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
7496  032454  104401  005532                  TYPE    ,SCRACH         ;ECHO CHARACTER TO BE DELETED
7497  032460  000710                          BR      2$              ;GO READ ANOTHER CHARACTER
7498  032462  005701                  14$:    TST     R1              ;CHECK THE RUB-OUT INDICATOR
7499  032464  001403                          BEQ     16$             ;BR IF INDICATOR IS NOT SET
7500  032466  104401  013415                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH
7501  032472  005001                          CLR     R1              ;CLEAR THE RUB-OUT INDICATOR
7502                                  ;CHECK FOR CARRIAGE RETURN
7503  032474  122702  000015          16$:    CMPB    #015,R2         ;SEE IF <CR> TYPED
7504  032500  001426                          BEQ     19$             ;BR IF <CR>
7505                                  ;HANDLE POSSIBLE DIGIT
7506  032502  005037  005532                  CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
```

```
032506  110237  005532                    MOVB   R2,SCRACH          ;GET THIS CHARACTER INTO BUFFER
032512  104401  005532                    TYPE   ,SCRACH            ;ECHO THE CHARACTER TYPED
032516  110260  005262                    MOVB   R2,BUFFO(R0)       ;PUT CHARACTER INTO BUFFER
032522  005200                            INC    R0                 ;INCREMENT CHARACTER COUNTER
032524  022700  000120                    CMP    #80.,R0            ;SEE IF TOO MANY CHARACTERS TYPED
032530  001264                            BNE    2$                 ;BR IF NOT TOO MANY
032532  104401  001315                    TYPE   ,$CRLF             ;TYPE <CR> AND <LF>
032536  112760  000000  005262            MOVB   #0,BUFFO(R0)     ;PUT TERMINATING NULL INTO BUFFER
032544  104401  005262                    TYPE   ,BUFFO             ;ECHO INPUT STRING
032550  104401  001314                    TYPE   ,$QUES             ;TYPE <?>,<CR>,<LF>
032554  000703                            BR     7$                 ;TAKE ERROR EXIT FROM RDCHRS
032556  104401  001315          19$:      TYPE   ,$CRLF             ;TYPE <CR>,<LF>
032562  112760  000000  005262            MOVB   #0,BUFFO(R0)       ;PUT TERMINATING NULL INTO BUFFER
032570  062766  000006  000004            ADD    #6,4(SP)           ;FIX UP RETURN ADDRESS
032576  012602                  24$:      MOV    (SP)+,R2           ;RESTORE R2
032600  012601                            MOV    (SP)+,R1           ;RESTORE R1
032602  000207                            RTS    PC                 ;SUBROUTINE EXIT



                                ;;*******************************************************
                                .SBTTL  TYPTST - TYPE CURRENT TEST AND ITERATION NUMBER
                                ;*THIS SUBROUTINE TYPES : "XX     YYYYYY", WITH NO <CR>
                                ;*OR <LF>. XX IS THE NUMBER OF THE CURRENT TEST AND
                                ;*YYYYYY IS THE NUMBER OF ITERATIONS FOR THAT TEST.
                                ;*R1 MUST CONTAIN THE INDEX INTO THE TEST LIST FOR
                                ;*THE CURRENT TEST.
                                ;* CALL -    JSR    PC,TYPTST
                                ;;*******************************************************

032604  104401  013426          TYPTST:   TYPE   ,SPACE1            ;TYPE A SPACE
032610  010146                            MOV    R1,-(SP)           ;PUT INDEX ONTO STACK
032612  006216                            ASR    (SP)               ;DIVIDE BY 2
032614  005216                            INC    (SP)               ;INCREMENT TO GET TEST NO.
032616  104403                            TYPOS                     ;TYPE TEST NO. IN OCTAL
032620     002                            .BYTE  2                  ;TYPE 2 DIGITS
032621     000                            .BYTE  0                  ;SUPPRESS LEADING ZEROS
032622  104401  013421                    TYPE   ,SPACE6            ;TYPE 6 SPACES
032626  016146  005616                    MOV    TSTLST(R1),-(SP)   ;PUT CURRENT ITERATION NO. ONTO STACK
032632  104403                            TYPOS                     ;TYPE ITERATION NO. IN OCTAL
032634     006                            .BYTE  6                  ;TYPE 6 DIGITS
032635     000                            .BYTE  0                  ;SUPPRESS LEADING ZEROS
032636  000207                            RTS    PC                 ;RETURN


                                ;;*******************************************************
                                .SBTTL  TYPPRM - TYPE CURRENT PARAMETER
                                ;*THIS SUBROUTINE TYPES THE LINE : XX=YYYYYYYY ,
                                ;*WHERE XX IS THE CURRENT PARAMETER MNEMONIC, AND
                                ;*YYYYYYYY IS ITS VALUE IN OCTAL DIGITS, WITH LEADING
                                ;*ZEROS SUPPRESSED. NOTE: <CR> AND <LF> ARE NOT DONE.
                                ;*ON ENTRY, R1 MUST CONTAIN THE INDEX INTO THE
                                ;*PARAMETER TABLES, FOR THE CURRENT PARAMETER.
                                ;* CALL -      JSR     PC,TYPPRM
                                ;;*******************************************************
```

```
 7563  032640  016137  005776  013430  TYPPRM: MOV     PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
 7564  032646  104401  013430                  TYPE    ,PRMBUF           ;TYPE "XX="
 7565  032652  016137  005646  005466          MOV     PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
 7566  032660  005037  005470                  CLR     HIGOCT            ;CLEAR HIGH BINARY BITS
 7567  032664  022761  040515  005776          CMP     #"MA,PRMNEM(R1)   ;SEE IF PARAMETER IS (MA)
 7568  032672  001003                          BNE     1$                ;BR IF NOT (MA)
 7569  032674  016137  005650  005470          MOV     PRMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
 7570  032702  012746  005466          1$:     MOV     #LOWOCT,-(SP)     ;PUT ADDRESS OF BINARY VALUE ON STACK
 7571  032706  004737  053752                  JSR     PC,@#$OB2O        ;CONVERT BINARY TO OCTAL ASCII
 7572  032712  004737  054266                  JSR     PC,@#$SUPRS       ;TYPE YYYYYYYY, SUPPRESSING LEADING O'S
 7573  032716  000207                          RTS     PC                ;RETURN
 7574
 7575
 7576
 7577                                  ;;*****************************************************************
 7578                                  ;SBTTL  TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15
 7579                                  ;*THIS SUBROUTINE TYPES THE LINE : WORD XX = YYYYYY,
 7580                                  ;*WHERE XX IS THE NUMBER OF THE CURRENT WORD OF
 7581                                  ;*USER-DEFINED PATTERN 15, AND YYYYYY IS ITS 16-BIT
 7582                                  ;*VALUE, IN OCTAL DIGITS. <CR> AND <LF> ARE NOT TYPED.
 7583                                  ;*ON ENTRY, R1 MUST CONTAIN THE INDEX OF THE CURRENT
 7584                                  ;*WORD IN THE PATTERN 15 TABLE.
 7585                                  ;*  CALL -     JSR       PC,TYPPAT
 7586                                  ;;*****************************************************************
 7587
 7588  032720                          TYPPAT:
 7589  032720  104401  013434                  TYPE    ,WORDSP           ;TYPE "WORD "
 7590  032724  010146                          MOV     R1,-(SP)          ;PUT INDEX ONTO STACK
 7591  032726  006216                          ASR     (SP)              ;DIVIDE BY TWO FOR WORD NO.
 7592  032730  104403                          TYPOS                     ;GO TYPE WORD NUMBER
 7593  032732     002                          .BYTE   2                 ;DIGIT COUNT = 2 FOR TYPOC
 7594  032733     001                          .BYTE   1                 ;TELL TYPOS TO TYPE LEADING ZEROS
 7595  032734  104401  013442                  TYPE    ,EQUALS           ;TYPE " = "
 7596  032740  016146  006764                  MOV     PAT15(R1),-(SP)   ;GET BINARY VALUE OF THIS WORD
 7597  032744  104403                          TYPOS                     ;TYPE VALUE IN OCTAL
 7598  032746     006                          .BYTE   6                 ;TYPE 6 DIGITS
 7599  032747     001                          .BYTE   1                 ;TYPE LEADING ZEROS
 7600  032750  000207                          RTS     PC                ;RETURN
 7601
 7602
 7603
 7604                                  ;;*****************************************************************
 7605                                  ;SBTTL  MODP15 - MODIFY USER-DEFINED PATTERN 15
 7606                                  ;*THIS SUBROUTINE DETERMINES WHETHER OR NOT USER-DEFINED
 7607                                  ;*DATA PATTERN 15 SHOULD BE OPENED FOR MODIFICATION, AND
 7608                                  ;*IF SO, IT REQUESTS THE NEW VALUES AND CONTROLS THEIR
 7609                                  ;*INPUT ON THE TTY. IT CHECKS VALUES FOR VALIDITY, AND
 7610                                  ;*LOADS THE 16 WORDS INTO THE PATTERN 15 TABLE (PAT15).
 7611                                  ;*IF THE OPERATOR ANSWERS THE REQUEST FOR A PATTERN WORD
 7612                                  ;*WITH EXCLAMATION POINT (!) (CR), THE SUBROUTINE
 7613                                  ;*PROPAGATES THE LAST WORD TYPED INTO ALL REMAINING
 7614                                  ;*WORDS OF THE PATTERN 15 TABLE.
 7615                                  ;*  CALL -    JSR       PC,MODP15
 7616                                  ;;*****************************************************************
 7617
 7618  032752                          MODP15:
```

```
 7619  032752  010046                        MOV     R0,-(SP)        ;SAVE R0
 7620  032754  010146                        MOV     R1,-(SP)        ;SAVE R1
 7621  032756  010246                        MOV     R2,-(SP)        ;SAVE R2
 7622                              ;SEE IF PARAMETER IS PT
 7623  032760  022761  052120  005776        CMP     #"PT,PRMNEM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
 7624  032766  001127                        BNE     22$             ;BR IF NOT (PT)
 7625                              ;SEE IF PATTERN 15 IS SPECIFIED
 7626  032770  032737  100000  005666        BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
 7627  032776  001523                        BEQ     22$             ;BR IF NOT SPECIFIED
 7628                              ;SEE IF PATTERN 15 SHOULD BE MODIFIED
 7629  033000  104401  010754        4$:     TYPE    ,MDFY15         ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
 7630  033004  004737  032272                JSR     PC,RDCHRS       ;READ RESPONSE
 7631  033010  033256                        24$                     ;(↑C) RETURN ADDRESS
 7632  033012  033266                        26$                     ;(↑Z) RETURN ADDRESS
 7633  033014  033052                        8$                      ;(↑U) OR ERROR RETURN ADDRESS
 7634  033016  005700                        TST     R0              ;SEE IF NULL INPUT
 7635  033020  001512                        BEQ     22$             ;BR IF MODIFICATION NOT REQUESTED
 7636  033022  022737  000115  005262        CMP     #'M,BUFFO       ;SEE IF (M) TYPED
 7637  033030  001405                        BEQ     6$              ;BR IF MODIFICATION REQUESTED
 7638  033032  104401  005262                TYPE    ,BUFFO          ;ECHO BAD INPUT
 7639  033036  104401  001314                TYPE    ,$QUES
 7640  033042  000756                        BR      4$              ;GO ASK AGAIN
 7641                              ;MODIFY PATTERN 15
 7642  033044  104401  010724        6$:     TYPE    ,SELP15         ;TYPE "MODIFY PATTERN 15"
 7643  033050  005001                        CLR     R1              ;INITIALIZE WORD INDEX
 7644  033052  004737  032720        8$:     JSR     PC,TYPPAT       ;TYPE CURRENT WORD AND VALUE
 7645  033056  104401  013426                TYPE    ,SPACE1         ;TYPE A SPACE
 7646  033062  104401  013446                TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
 7647                              ;READ AND CHECK INPUT, IF ANY
 7648  033066  004737  032272                JSR     PC,RDCHRS       ;READ NEW DATA PATTERN WORD
 7649  033072  033256                        24$                     ;(↑C) RETURN ADDRESS FOR RDCHRS
 7650  033074  033266                        26$                     ;(↑Z) RETURN ADDRESS FOR RDCHRS
 7651  033076  033052                        8$                      ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
 7652  033100  005700                        TST     R0              ;SEE IF ANY INPUT
 7653  033102  001006                        BNE     12$             ;BR IF ANY INPUT
 7654  033104  062701  000002        10$:    ADD     #2,R1           ;INCREMENT WORD INDEX
 7655  033110  022701  000040                CMP     #32.,R1         ;SEE IF ALL DONE
 7656  033114  001454                        BEQ     22$             ;BR IF DONE, TO RETURN
 7657  033116  000755                        BR      8$              ;CONTINUE WITH NEXT WORD
 7658  033120  022737  000041  005262  12$:  CMP     #'!,BUFFO       ;SEE IF (!) TYPED
 7659  033126  001431                        BEQ     16$             ;BR TO PROPAGATE CURRENT VALUE
 7660  033130  005002                        CLR     R2              ;INIT. (!) INDICATOR
 7661  033132  122760  000041  005261        CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
 7662  033140  001004                        BNE     14$             ;BR IF NOT (!)
 7663  033142  105060  005261                CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
 7664  033146  005300                        DEC     R0              ;DECREMENT CHAR COUNT
 7665  033150  005202                        INC     R2              ;SET (!) INDICATOR
 7666  033152  022700  000006        14$:    CMP     #6,R0           ;SEE HOW MANY CHARS NOW
 7667  033156  002426                        BLT     20$             ;BR IF TOO MANY
 7668  033160  012746  005262                MOV     #BUFFO,-(SP)    ;GET BUF. ADDR. ON STACK FOR OCTBIN
 7669  033164  004737  053014                JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
 7670  033170  033234                        20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
 7671  033172  012600                        MOV     (SP)+,R0        ;GET BINARY VALUE
 7672  033174  005737  053146                TST     $HIOCT          ;SEE IF VALUE EXCEEDS 16 BITS
 7673  033200  001015                        BNE     20$             ;BR TO ECHO BAD INPUT
 7674  033202  010061  006764                MOV     R0,PAT15(R1)    ;PUT NEW WORD VALUE INTO TABLE
```

```
 7675  033206  005702                        TST     R2              ;SEE IF (!) WAS TYPED
 7676  033210  001735                        BEQ     10$             ;BR IF (!) WAS NOT TYPED
 7677                                ;PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE
 7678  033212  022701  000036          16$:  CMP     #30.,R1         ;SEE IF ALL DONE YET
 7679  033216  001413                        BEQ     22$             ;BR IF DONE
 7680  033220  016161  006764  006766        MOV     PAT15(R1),PAT15+2(R1) ;PROPAGATE TO NEXT WORD
 7681  033226  062701  000002                ADD     #2,R1           ;INCREMENT WORD INDEX
 7682  033232  000767                        BR      16$             ;LOOP UNTIL DONE
 7683                                ;ECHO BAD INPUT
 7684  033234  104401  005262          20$:  TYPE    ,BUFFO          ;ECHO BAD INPUT
 7685  033240  104401  001314                TYPE    ,$QUES          ;TYPE '?' AND <CR>,<LF>
 7686  033244  000702                        BR      8$              ;BR TO ASK AGAIN
 7687                                ;NORMAL RETURN
 7688  033246  012602                  22$:  MOV     (SP)+,R2        ;RESTORE R2
 7689  033250  012601                        MOV     (SP)+,R1        ;RESTORE R1
 7690  033252  012600                        MOV     (SP)+,R0        ;RESTORE R0
 7691  033254  000207                        RTS     PC              ;RETURN
 7692                                ;(↑C) RETURN
 7693  033256  012766  014532  000006  24$:  MOV     #DRVTST,6(SP)   ;PC FOR (↑C) RETURN
 7694  033264  000770                        BR      22$             ;BR TO RETURN
 7695                                ;(↑Z) RETURN
 7696  033266  012766  016156  000006  26$:  MOV     #ASKMDE,6(SP)   ;PC FOR (↑Z) RETURN
 7697  033274  000764                        BR      22$             ;BR TO RETURN
 7698
 7699
 7700
 7701                                ;;*****************************************************************
 7702                                ;SBTTL  CHKPRM - CHECK VALUE OF INPUT PARAMETER
 7703                                ;*THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
 7704                                ;*HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
 7705                                ;*LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
 7706                                ;*INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
 7707                                ;*TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
 7708                                ;*IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
 7709                                ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
 7710                                ;*THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
 7711                                ;*VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
 7712                                ;* CALL -      JSR     PC,CHKPRM
 7713                                ;*           <ERROR RETURN ADDRESS>
 7714                                ;*           RETURN
 7715                                ;;*****************************************************************
 7716
 7717  033276  104407                CHKPRM: SAVREG                  ;SAVE R0-R5
 7718  033300  010102                        MOV     R1,R2           ;GET COPY OF INDEX
 7719  033302  006302                        ASL     R2              ;DOUBLE IT
 7720  033304  022761  040515  005776        CMP     #"MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
 7721  033312  001422                        BEQ     20$             ;BR IF (MA)
 7722  033314  005737  005470                TST     HIGOCT          ;SEE IF HIGH BITS = 0
 7723  033320  001403                        BEQ     18$             ;BR IF ZERO
 7724  033322  017616  000000          16$:  MOV     @(SP),(SP)      ;GET ERROR RETURN PC
 7725  033326  000475                        BR      30$             ;GO TO RETURN
 7726                                ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
 7727  033330  023762  005466  005722  18$:  CMP     LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
 7728  033336  103771                        BLO     16$             ;BR IF INPUT VALUE TOO SMALL
 7729  033340  023762  005466  005724        CMP     LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
 7730  033346  101365                        BHI     16$             ;BR IF INPUT VALUE IS TOO LARGE
```

```
                                        ;UPDATE 16-BIT PARAMETER VALUE IN LIST
033350  013761  005466  005646            MOV     LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
033356  000457                            BR      28$             ;BR TO RETURN
                                        ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
033360  032737  000001  005466  20$:     BIT     #BIT0,LOWOCT    ;SEE IF MA IS ODD
033366  001355                            BNE     16$             ;BR IF MA IS ODD
033370  023762  005470  005724            CMP     HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
033376  103751                            BLO     16$             ;BR IF HIGH WORD IS TOO SMALL
033400  001004                            BNE     24$             ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
033402  023762  005466  005722            CMP     LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
033410  103744                            BLO     16$             ;BR IF LOW WORD IS TOO SMALL
033412  023762  005470  005730  24$:     CMP     HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
033420  101340                            BHI     16$             ;BR IF HIGH WORD TOO BIG
033422  001004                            BNE     26$             ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
033424  023762  005466  005726            CMP     LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
033432  101333                            BHI     16$             ;BR IF LOW WORD IS TOO LARGE
                                        ;UPDATE 32-BIT PARAMETER VALUE IN LIST
033434  013761  005466  005646  26$:     MOV     LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
033442  013761  005470  005650            MOV     HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
                                        ;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
033450  004737  033526                    JSR     PC,MXWRDC       ;GET WORD COUNT IN R1-R0
033454  104401  010455                    TYPE    ,MAWRDC         ;TYPE "MAX WORD COUNT = "
033460  010037  003166                    MOV     R0,SUML01
033464  010137  003170                    MOV     R1,SUMH11
033470  012746  003166                    MOV     #SUML01,-(SP)   ;PUT POINTER ON STACK
033474  004737  053752                    JSR     PC,$DB2O        ;CONVERT TO OCTAL
033500  004737  054266                    JSR     PC,$SUPRS       ;TYPE IT
033504  104401  001315                    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
033510  062766  000002  000014            ADD     #2,14(SP)       ;INCREMENT R1 INDEX
033516  062716  000002            28$:    ADD     #2,(SP) ;GET NORMAL RETURN PC
033522  104410                    30$:    RESREG                  ;RESTORE R0-R5
033524  000207                            RTS     PC              ;RETURN


;;*****************************************************************
;*MXWRDC - GET MAX WORD COUNT FOR CURRENT MEM LIMITS AND CURRENT MA.
;*AND LEAVE THE WORD COUNT IN R1-R0. NO REGISTERS ARE SAVED.
;;*****************************************************************
033526  013700  005572            MXWRDC: MOV     MAHILM,R0       ;GET LO BITS OF MEM LIM
033532  013701  005574                    MOV     MAHILM+2,R1     ;HI BITS OF MEM LIM
033536  163700  005576                    SUB     MA,R0           ;SUBTRACT MA FROM MAHILM
033542  005601                            SBC     R1
033544  163701  005600                    SUB     MA+2,R1
033550  100413                            BMI     27$             ;IF NEGATIVE, RETURN
033552  000241                            CLC                     ;DIVIDE BY 2 TO GET WORDS
033554  006001                            ROR     R1
033556  006000                            ROR     R0
033560  062700  000001                    ADD     #1,R0           ;INCREMENT BY 1 WORD
033564  005501                            ADC     R1
033566  005701                            TST     R1
033570  001403                            BEQ     27$             ;BR IF WORD COUNT < 65,536 DEC
033572  005000                            CLR     R0              ;MAKE WORD COUNT = 65,536
033574  012701  000001                    MOV     #1,R1
033600  000207                    27$:    RTS     PC              ;RETURN
```

```
7787
7788
7789                                    ;;*****************************************************************
7790                                    .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
7791                                    ;*THIS SUBROUTINE TYPES "DRIVE SER. NO.  XXX" (IN DECIMAL), WITH LEADING
7792                                    ;*ZEROS SUPPRESSED.
7793                                    ;;*****************************************************************
7794  033622  104407                    DRVSER: SAVREG                          ;SAVE R0-R5
7795  033604  004737  030776                    JSR     PC,INITSS               ;CLEAR S.S. AND PARAMETERS
7796  033610  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
7797  033616  004737  040774                    JSR     PC,DRVCAL               ;READ STATUS OF THIS DRIVE
7798  033622  104401  011173                    TYPE    ,DRIV                   ;TYPE "DRIVE"
7799  033626  104401  011211                    TYPE    ,SERNM                  ;TYPE " SER. NO.  "
7800  033632  016501  000054                    MOV     P.A11(R5),R1            ;GET "A" STATUS BYTE 11
7801  033636  012704  054054                    MOV     #$OCTVL,R4              ;GET ADDR OF CHAR BUFFER
7802  033642  010446                            MOV     R4,-(SP)                ;STORE IT ON STACK FOR $SUPRS
7803  033644  012703  000003                    MOV     #3,R3                   ;INIT CHAR COUNT
7804  033650  006101                            ROL     R1                      ;INITIALIZE BIT POSITIONS
7805  033652  006101                            ROL     R1
7806  033654  006101                    4$:     ROL     R1                      ;GET NEXT 4 BITS
7807  033656  006101                            ROL     R1
7808  033660  006101                            ROL     R1
7809  033662  006101                            ROL     R1
7810  033664  010100                            MOV     R1,R0                   ;GET A WORKING COPY
7811  033666  042700  177760                    BIC     #177760,R0              ;CLEAR ALL BUT LOW 4 BITS
7812  033672  052700  000060                    BIS     #'0,R0                  ;CONVERT A DIGIT TO ASCII
7813  033676  110024                            MOVB    R0,(R4)+                ;PUT ASCII DIGIT INTO CHAR BUFFER
7814  033700  005303                            DEC     R3                      ;DECREMENT CHAR COUNT
7815  033702  001364                            BNE     4$                      ;BR IF NOT 3 CHARS YET
7816  033704  105014                            CLRB    (R4)                    ;INSERT NULL TERMINATOR
7817  033706  004737  054266                    JSR     PC,@#$SUPRS             ;TYPE DRIVE SER. NUMBER
7818  033712  104401  001315                    TYPE    ,$CRLF                  ;TYPE <CR> AND <LF>
7819  033716  104410                            RESREG                          ;RESTORE R0-R5
7820  033720  000207                            RTS     PC                      ;RETURN
7821
7822
7823
7824                                    ;;*****************************************************************
7825                                    .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
7826                                    ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXXX" (IN OCTAL),
7827                                    ;*WITH LEADING ZEROS SUPPRESSED.
7828                                    ;;*****************************************************************
7829  033722  004737  030776            CRTSER: JSR     PC,INITSS               ;CLEAR S.S. AND PARAMETERS
7830  033726  105065  000007                    CLRB    P.CS1H(R5)              ;SET 22-SECTOR FORMAT
7831  033732  105737  003115                    TSTB    FORMAT                  ;CHECK THE ACTUAL FORMAT
7832  033736  001402                            BEQ     4$                      ;BR IF 22 SECTORS
7833  033740  105265  000004                    INCB    P.SECT(R5)              ;IF 20 SECTORS, READ SECTOR 1
7834  033744  112765  000121  000001    4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
7835  033752  012765  000632  000002            MOV     #LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)
7836  033760  112765  000002  000005            MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
7837  033766  012765  064640  000010            MOV     #RWBUF,P.BAL0(R5) ;SET READ BUFFER ADDRESS
7838  033774  012765  177776  000012            MOV     #-2,P.WC(R5)            ;SET WORD COUNT TO READ 2 WORDS
7839  034002  004737  040774                    JSR     PC,DRVCAL               ;READ SERIAL NO. IN BSF
7840  034006  104401  011202                    TYPE    ,CART                   ;TYPE   "CART."
7841  034012  104401  011211                    TYPE    ,SERNM                  ;TYPE "SER. NO.  "
7842  034016  012746  064640                    MOV     #RWBUF,-(SP)            ;GET POINTER FOR $DB2O
```

```
 7843  034022  004737  053752                  JSR     PC,@#$CB2O      ;CONVERT BINARY TO OCTAL
 7844  034026  004737  054266                  JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
 7845  034032  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7846  034036  000207                          RTS     PC              ;RETURN
 7847
 7848
 7849
 7850                                  ;;*****************************************************************
                                       .SBTTL  STALL - STALL FOR ST UNIT STALL TIMES
                                       ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
                                       ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
                                       ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
                                       ;* CALL -   JSR     PC,STALL
                                       ;;*****************************************************************
 7858  034040  010046                  STALL:  MOV     R0,-(SP)        ;SAVE R0
 7859  034042  010146                          MOV     R1,-(SP)        ;SAVE R1
 7860  034044  032777  000400  145066          BIT     #BIT08,@SWR     ;APPLY RANDOM STALL ?
 7861  034052  001407                          BEQ     1$              ;BR IF NOT RANDOM
 7862  034054  004737  053150                  JSR     PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
 7863  034060  013700  053250                  MOV     $LONUM,R0       ;GET IT INTO R0
 7864  034064  006200                          ASR     R0              ;SCALE IT DOWN
 7865  034066  006200                          ASR     R0
 7866  034070  000403                          BR      2$              ;GO STALL WITH RANDOM NO.
 7867  034072  013700  005502          1$:     MOV     STALLS,R0       ;GET REQUESTED NO. OF STALLS
 7868  034076  001406                          BEQ     6$              ;RETURN IF NO STALL REQUIRED
 7869  034100  012701  000016          2$:     MOV     #14.,R1         ;SET CONSTANT FOR 40 LS
 7870  034104  005301                  4$:     DEC     R1              ;INNER LOOP COUNTER
 7871  034106  001376                          BNE     4$              ;INNER LOOP BR
 7872  034110  005300                          DEC     R0              ;OUTER LOOP COUNTER
 7873  034112  001372                          BNE     2$              ;OUTER LOOP BR
 7874  034114  012601                  6$:     MOV     (SP)+,R1        ;RESTORE R1
 7875  034116  012600                          MOV     (SP)+,R0        ;RESTORE R0
 7876  034120  000207                          RTS     PC              ;RETURN
 7877
 7878
 7879
 7880                                  ;;*****************************************************************
                                       .SBTTL  RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
                                       ;*THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
                                       ;*ADDRESS, AND LEAVES IT IN "CYLNDR". IT REQUIRES THE SYSMAC
                                       ;*SUBROUTINE $RAND.
                                       ;* CALL -   JSR     PC,RNDADR
                                       ;;*****************************************************************
 7887  034122  004737  053150          RNDADR: JSR     PC,$RAND        ;GENERATE 2 16-BIT RANDOM NUMBERS
 7888  034126  013737  053250  005504          MOV     $LONUM,CYLNDR   ;GET A RANDOM NUMBER
 7889  034134  042737  177000  005504          BIC     #177000,CYLNDR  ;SCALE IT TO 9 BITS
 7890  034142  022737  000632  005504          CMP     #632,CYLNDR     ;SEE IF CYL IS TOO BIG
 7891  034150  002003                          BGE     2$              ;BR IF CYL IS OK
 7892  034152  042737  000400  005504          BIC     #BIT08,CYLNDR   ;SCALE DOWN TO A VALID CYLINDER
 7893  034160  000207                  2$:     RTS     PC              ;RETURN
 7894
 7895
 7896
 7897                                  ;;*****************************************************************
                                       ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF R0 INTO ALL
```

```
                                        ;*256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
                                        ;;**************************************************************
034162  104407                  LODSEC: SAVREG                          ;SAVE R0-R5
034164  012701  064640                  MOV     #RWBUF,R1               ;GET ADDRESS OF DATA BUF INTO R1
034170  010021                  2$:     MOV     R0,(R1)+                ;PUT WORD INTO BUFFER
034172  020127  065640                  CMP     R1,#RWBUF+512.          ;SEE IF 256 WORDS WRITTEN YET
034176  001374                          BNE     2$                      ;BR IF NOT DONE YET
034200  104410                          RESREG                          ;RESTORE R0-R5
034202  000207                          RTS     PC                      ;RETURN


                                        ;;**************************************************************
                                        ;* TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
                                        ;*FC, FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
                                        ;*TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
                                        ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.
                                        ;;**************************************************************
034204  104407                  TRKCHK: SAVREG                          ;SAVE R0-R5
034206  016546  000004                  MOV     P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
034212  105065  000005                  CLRB    P.TRCK(R5)              ;CLEAR THE TRACK NO.
                                        ;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
034216  116500  000005          2$:     MOVB    P.TRCK(R5),R0   ;GET TRACK NO. INTO R0
034222  062700  000100                  ADD     #100,R0                 ;ADD 100(OCT) TO TRACK NO.
034226  004737  034162                  JSR     PC,LODSEC               ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
034232  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
034240  004737  040774                  JSR     PC,DRVCAL               ;PERFORM THE WRITE CHECK
034244  105265  000005                  INCB    P.TRCK(R5)              ;INCREMENT THE TRACK NO.
034250  122765  000003  000005          CMPB    #3,P.TRCK(R5)   ;SEE IF DONE WITH ALL TRACKS
034256  001357                          BNE     2$                      ;BR IF NOT DONE YET
034260  012665  000004                  MOV     (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
034264  104410                          RESREG                          ;RESTORE R0-R5
034266  000207                          RTS     PC                      ;RETURN


                                        ;;**************************************************************
                                        ;*LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
                                        ;*INTO THE PATTERN 14 TABLE.
                                        ;;**************************************************************
034270  104407                  LODP14: SAVREG                          ;SAVE R0-R5
034272  012701  000010                  MOV     #8.,R1                  ;INIT LOOP COUNTER TO 8.
034276  012700  006724                  MOV     #PAT14,R0               ;GET ADDRESS OF PATTERN 14 BUFFER
034302  004737  053150          4$:     JSR     PC,$RAND                ;GENERATE 2 16-BIT RANDOM NUMBERS
034306  013720  053250                  MOV     $LONUM,(R0)+            ;PUT ONE NUMBER INTO PATTERN
034312  013720  053246                  MOV     $HINUM,(R0)+            ;PUT OTHER NO. INTO PATTERN
034316  005301                          DEC     R1                      ;SEE IF 16 WORDS LOADED YET
034320  001370                          BNE     4$                      ;BR IF NOT YET
034322  104410                          RESREG                          ;RESTORE R0-R5
034324  000207                          RTS     PC                      ;RETURN


                                        ;;**************************************************************
                                        .SBTTL  FINADR - COMPUTE FINAL PACK ADDRESS
                                        ;*THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
```

```
                                ;*COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
                                ;*WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
                                ;*P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
                                ;*PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
                                ;*THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
                                ;*DATA MISCOMPARE.
                                ;;*****************************************************************
034326  104407                  FINADR: SAVREG                          ;SAVE R0-R5
034330  013746  005570                  MOV     LASTWC,-(SP)            ;STORE WORD COUNT
034334  005416                          NEG     (SP)                    ;MAKE IT POSITIVE
034336  005046                  18$:    CLR     -(SP)                   ;MAKE ROOM ON STACK
034340  116616  000003                  MOVB    3(SP),(SP)              ;STORE NO. OF SECTORS TRANSFERRED
034344  005066  000002                  CLR     2(SP)                   ;CLEAR LOCATION ON STACK
034350  116566  000004  000002          MOVB    P.SECT(R5),2(SP)        ;STORE STARTING SECTOR
034356  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL SECTOR ADDRESS
034362  005066  000002                  CLR     2(SP)                   ;CLEAR NO. OF TRACKS TRANSFERRED
034366  012700  000026                  MOV     #22.,R0                 ;SET FOR 22 SECTORS
034372  105737  003115                  TSTB    FORMAT                  ;DETERMINE THE FORMAT
034376  001402                          BEQ     19$                     ;BR IF 22 SECTORS
034400  012700  000024                  MOV     #20.,R0                 ;SET FOR 20 SECTORS
034404  020016                  19$:    CMP     R0,(SP)                 ;CHECK FOR SECTOR OVERFLOW
034406  101004                          BHI     20$                     ;NO, CHECK IF SECTOR CORRECT
034410  160016                          SUB     R0,(SP)                 ;DECREMENT SECTOR COUNT BY 20 OR 22
034412  005266  000002                  INC     2(SP)                   ;INCREMENT TRACKS TRANSFERRED
034416  000772                          BR      19$                     ;CHECK FOR SECTOR OVERFLOW
034420  112637  005567          20$:    MOVB    (SP)+,FINSEC            ;STORE FINAL SECTOR
034424  005046                          CLR     -(SP)                   ;MAKE ROOM FOR TRACKS TRANSFERRED
034426  116516  000005                  MOVB    P.TRCK(R5),(SP)         ;STORE STARTING TRACK
034432  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL TRACK ADDRESS
034436  005066  000002                  CLR     2(SP)                   ;CLEAR FINAL CYLINDER
034442  122716  000003          21$:    CMPB    #3,(SP)                 ;CHECK FOR TRACK OVERFLOW
034446  101005                          BHI     22$                     ;NO, CHECK FINAL TRACK
034450  162716  000003                  SUB     #3,(SP)                 ;DECREMENT TRACK COUNT BY 3
034454  005266  000002                  INC     2(SP)                   ;INCR CYL COUNT
034460  000770                          BR      21$                     ;CHECK FOR TRACK OVERFLOW
034462  112637  005566          22$:    MOVB    (SP)+,FINTRK            ;STORE FINAL TRACK
034466  066516  000002                  ADD     P.CYLN(R5),(SP)         ;CALCULATE FINAL CYLINDER
034472  011637  005564                  MOV     (SP),FINCYL             ;STORE FINAL  CYLINDER
034476  005726                          TST     (SP)+                   ;CLEAN OFF STACK
034500  104410                          RESREG                          ;RESTORE R0-R5
034502  000207                          RTS     PC                      ;RETURN
                                ;;*****************************************************************
                                .SBTTL  LODBUF - LOAD THE READ/WRITE DATA BUFFER
                                ;*THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
                                ;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
                                ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
                                ;*OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
                                ;*IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
                                ;*OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
                                ;*ENTRY, ARE LOADED INTO THE BUFFER.
                                ;;*****************************************************************
034504  104407                  LODBUF: SAVREG                          ;SAVE R0-R5
```

```
 8011  034506  013702  005562                MOV     WDSXFR,R2       ;GET NO. OF WORDS
 8012  034512  005737  005534                TST     PATRN           ;SEE IF QUICK VERIFY DATA TEST DESIRED
 8013  034516  001007                        BNE     3$              ;BR IF NOT QUICK VERIFY
 8014  034520  012700  006024                MOV     #PAT00,R0       ;SET DATA PATTERN STARTING ADDRESS
 8015  034524  012746  000400                MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
 8016  034530  012701  064640                MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
 8017  034534  000463                        BR      30$             ;PROCEED
 8018  034536  005000              3$:       CLR     R0              ;INIT PATTERN NUMBER
 8019  034540  032701  000001        4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
 8020  034544  001003                        BNE     6$              ;BR IF THIS BIT IS SET
 8021  034546  005200                        INC     R0              ;INCREMENT PATTERN NO.
 8022  034550  006201                        ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
 8023  034552  000772                        BR      4$              ;BR TO CHECK NEXT BIT
 8024  034554  006300              6$:       ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
 8025  034556  006300                        ASL     R0
 8026  034560  006300                        ASL     R0
 8027  034562  006300                        ASL     R0
 8028  034564  006300                        ASL     R0
 8029  034566  062700  006024                ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
 8030  034572  012746  000020                MOV     #16.,-(SP)      ;SET PATTERN WORD COUNT
 8031  034576  013701  005602                MOV     PMA,R1          ;SET BUFFER ADDRESS
 8032  034602  005737  056246                TST     $KT11           ;SEE IF MEM MGT PRESENT
 8033  034606  100036                        BPL     30$             ;BR IF NOT PRESENT
 8034                          ;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
 8035  034610  013737  003176  172354        MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
 8036  034616  052737  000001  177572        BIS     #BIT0,@#SR0     ;TURN ON MEMORY MANAGEMENT
 8037  034624  042701  160000                BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
 8038  034630  052701  140000                BIS     #140000,R1
 8039  034634  010003              22$:      MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8040  034636  011604                        MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8041  034640  012321              24$:      MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8042  034642  032701  020000                BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
 8043  034646  001405                        BEQ     26$             ;BR IF NO OVERFLOW
 8044  034650  062737  000200  172354        ADD     #200,@#KIPAR6   ;INCREMENT PAR BY 4K FOR NEW PAGE
 8045  034656  042701  020000                BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 8046  034662  005302              26$:      DEC     R2              ;DECREMENT WORD COUNTER
 8047  034664  001403                        BEQ     28$             ;BR IF ALL DONE
 8048  034666  005304                        DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8049  034670  001363                        BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
 8050  034672  000760                        BR      22$             ;BR TO REPEAT THE PATTERN
 8051  034674  042737  000001  177572  28$:  BIC     #BIT0,@#SR0     ;DISABLE MEMORY MANAGEMENT
 8052  034702  000410                        BR      44$             ;GO TO RETURN
 8053                          ;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE
 8054  034704  010003              30$:      MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8055  034706  011604                        MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8056  034710  012321              34$:      MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8057  034712  005302                        DEC     R2              ;DECREMENT WORD COUNTER
 8058  034714  001403                        BEQ     44$             ;BR IF ALL DONE
 8059  034716  005304                        DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8060  034720  001373                        BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
 8061  034722  000770                        BR      30$             ;BR TO REPEAT THE PATTERN
 8062  034724  005726              44$:      TST     (SP)+           ;POP THE STACK
 8063  034726  104410                        RESREG                  ;RESTORE R0-R5
 8064  034730  000207                        RTS     PC              ;RETURN
 8065
 8066
```

```
 8067
 8068                                   ;:*****************************************************************
 8069                                   .SBTTL  CMPBUF - SOFTWARE COMPARE DATA
 8070                                   ;*THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
 8071                                   ;*TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
 8072                                   ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
 8073                                   ;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
 8074                                   ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
 8075                                   ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
 8076                                   ;:*****************************************************************
 8077  034732  104407                  CMPBUF: SAVREG                          ;SAVE R0-R5
 8078  034734  112737  000040  063157          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
 8079  034742  012737  000007  064532          MOV     #7,DF25+2
 8080  034750  013702  005562                  MOV     WDSXFR,R2       ;SET NO. OF WORDS
 8081  034754  005037  005532                  CLR     SCRACH          ;CLEAR COMPARE ERROR COUNT
 8082  034760  004737  042206                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
 8083  034764  005737  005534                  TST     PATRN           ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
 8084  034770  001017                          BNE     3$              ;BR IF NOT
 8085  034772  012700  006024                  MOV     #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
 8086  034776  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
 8087  035002  012701  064640                  MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
 8088  035006  000466                          BR      30$             ;PROCEED
 8089  035010  005000          3$:     CLR     R0              ;INIT PATTERN NO.
 8090  035012  032701  000001  4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
 8091  035016  001003                          BNE     6$              ;BR IF THIS BIT IS SET
 8092  035020  005200                          INC     R0              ;INCREMENT PATTERN NUMBER
 8093  035022  006201                          ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
 8094  035024  000772                          BR      4$              ;BR TO CHECK NEXT BIT
 8095  035026  006300          6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
 8096  035030  006300                          ASL     R0
 8097  035032  006300                          ASL     R0
 8098  035034  006300                          ASL     R0
 8099  035036  006300                          ASL     R0
 8100  035040  062700  006024                  ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
 8101  035044  012746  000020                  MOV     #16.,-(SP)      ;PATTERN WORD COUNT
 8102  035050  013701  005602                  MOV     PMA,R1          ;SET BUFFER ADDRESS
 8103  035054  005737  056246                  TST     $KT11           ;SEE IF MEM MGT PRESENT
 8104  035060  100041                          BPL     30$             ;BR IF NOT PRESENT
 8105                                   ;COMPARE LOOP FOR MEM MGT STARTS HERE
 8106  035062  013737  003176  172354          MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
 8107  035070  052737  000001  177572          BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
 8108  035076  042701  160000                  BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
 8109  035102  052701  140000                  BIS     #140000,R1
 8110  035106  010003          22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8111  035110  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8112  035112  022321          24$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
 8113  035114  001404                          BEQ     25$             ;BR IF NO ERROR
 8114  035116  013737  172354  001216          MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
 8115  035124  000432                          BR      35$             ;GO HANDLE ERROR
 8116  035126  032701  020000          25$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
 8117  035132  001405                          BEQ     26$             ;BR IF NO OVERFLOW
 8118  035134  062737  000200  172354          ADD     #200,@#KIPAR6   ;INCR PAR BY 4K FOR NEW PAGE
 8119  035142  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 8120  035146  005302          26$:    DEC     R2              ;DECREMENT WORD COUNTER
 8121  035150  001002                          BNE     27$             ;BR IF NOT ALL DONE YET
 8122  035152  000137  035616                  JMP     54$             ;ALL DONE - GET OUT
```

```
 8123  035156  005304                  27$:    DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8124  035160  001354                          BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
 8125  035162  000751                          BR      22$             ;BR TO REPEAT THE PATTERN
 8126                                  ;COMPARE LOOP FOR NO MEM MGT STARTS HERE
 8127  035164  010003                  30$:    MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
 8128  035166  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8129  035170  022321                  34$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
 8130  035172  001002                          BNE     31$             ;BR IF COMPARE ERROR
 8131  035174  000137  035576                  JMP     40$             ;JUMP IF DATA COMPARES OK
 8132  035200  105037  063157          31$:    CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
 8133  035204  012737  000005  064532          MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
 8134                                  ;COMMON COMPARE ERROR HANDLER
 8135  035212  010237  001202          35$:    MOV     R2,$REG10       ;GET WORD NO.
 8136  035216  163737  005562  001202          SUB     WDSXFR,$REG10
 8137  035224  013737  001202  005570          MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
 8138  035232  004737  034326                  JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
 8139  035236  104407                          SAVREG                  ;SAVE R0-R5
 8140  035240  013700  005564                  MOV     FINCYL,R0       ;GET CYL
 8141  035244  113701  005566                  MOVB    FINTRK,R1       ;GET TRACK
 8142  035250  113702  005567                  MOVB    FINSEC,R2       ;GET SECTOR
 8143  035254  004737  040670                  JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
 8144  035260  104410                          RESREG                  ;RESTORE R0-R5
 8145  035262  032737  001000  005474          BIT     #BADSEC,RECODE
 8146  035270  001132                          BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
 8147  035272  005737  005532                  TST     SCRACH          ;CHECK THE ERROR COUNT
 8148  035276  001024                          BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
 8149  035300  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 8150  035304  001411                          BEQ     46$             ;BR IF NOT
 8151  035306  013737  005260  001174          MOV     CRMPH0,$REG5    ;GET CURRENT MAP REG 0
 8152  035314  013737  005256  001176          MOV     CRMPL0,$REG6
 8153  035322  104116                          ERROR   116             ;DATA MISCOMPARE (11/70)
 8154  035324  104117                          ERROR   117
 8155  035326  000401                          BR      48$
 8156  035330  104034                  46$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
 8157  035332  012737  177777  001174  48$:    MOV     #-1,$REG5       ;INIT CYL NO.
 8158  035340  005037  001176                  CLR     $REG6
 8159  035344  005037  001200                  CLR     $REG7
 8160  035350  005237  005532          36$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
 8161  035354  032777  000001  143556          BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
 8162  035362  001004                          BNE     38$             ;BR TO REPORT ALL ERRORS
 8163  035364  022737  000012  005532          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
 8164  035372  002511                          BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
 8165  035374  023737  001174  005564  38$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
 8166  035402  001010                          BNE     42$             ;BR IF YES
 8167  035404  123737  001176  005566          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
 8168  035412  001004                          BNE     42$             ;BR IF YES
 8169  035414  123737  001200  005567          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
 8170  035422  001412                          BEQ     44$             ;BR IF SAME PACK ADDRESS
 8171  035424  013737  005564  001174  42$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
 8172  035432  113737  005566  001176          MOVB    FINTRK,$REG6
 8173  035440  113737  005567  001200          MOVB    FINSEC,$REG7
 8174  035446  104115                          ERROR   115             ;TYPE NEW PACK ADDRESS
 8175  035450  005437  001202          44$:    NEG     $REG10          ;GET WORD NO.
 8176  035454  016337  177776  001204          MOV     -2(R3),$REG11   ;GET GOOD DATA
 8177  035462  016137  177776  001206          MOV     -2(R1),$REG12   ;GET BAD DATA
 8178  035470  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
```

```
 8179  035476  005037  001210                  CLR     $REG13
 8180  035502  006137  001212                  ROL     $REG14
 8181  035506  006137  001210                  ROL     $REG13
 8182  035512  063737  005602  001212          ADD     PMA,$REG14
 8183  035520  005537  001210                  ADC     $REG13
 8184  035524  063737  005604  001210          ADD     PMA+2,$REG13
 8185  035532  010137  001214                  MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
 8186  035536  162737  000002  001214          SUB     #2,$REG15
 8187  035544  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
 8188  035546  032777  000100  143364          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
 8189  035554  001020                          BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
 8190  035556  005737  005534          50$:    TST     PATRN           ;SEE IF DEFAULT DATA TEST
 8191  035562  001405                          BEQ     40$             ;BR IF YES
 8192  035564  005737  056246                  TST     $KT11           ;SEE IF MEM MGT PRESENT
 8193  035570  100002                          BPL     40$             ;BR IF NOT PRESENT
 8194  035572  000137  035126                  JMP     25$             ;GO HANDLE MEM MGT
 8195  035576  005302                  40$:    DEC     R2              ;DECREMENT WORD COUNTER
 8196  035600  001406                          BEQ     54$             ;BR IF ALL DONE
 8197  035602  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8198  035604  001002                          BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
 8199  035606  000137  035164                  JMP     30$             ;JUMP TO REPEAT THE PATTERN
 8200  035612  000137  035170          53$:    JMP     34$
 8201  035616  005737  056246          54$:    TST     $KT11           ;SEE IF MEM MGT PRESENT
 8202  035622  100003                          BPL     56$             ;BR IF NOT PRESENT
 8203  035624  042737  000001  177572          BIC     #BIT0,@#SR0     ;DISABLE MEM MGT
 8204  035632  005726                  56$:    TST     (SP)+           ;POP THE STACK
 8205  035634  104410                          RESREG                  ;RESTORE R0-R5
 8206  035636  000207                          RTS     PC              ;RETURN
 8207
 8208
 8209
 8210                                  ;;*****************************************************************
 8211                                  ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
 8212                                  ;*IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (↑Z)
 8213                                  ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
 8214                                  ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
 8215                                  ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
 8216                                  ;*NO ACTION IS TAKEN.
 8217                                  ;*  CALL -   JSR     PC,CTLOUT
 8218                                  ;*    OR -   CKEXIT
 8219                                  ;;*****************************************************************
 8220  035640  122737  000001  003110  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
 8221  035646  001077                          BNE     10$             ;BR IF NOT RUNNING TESTS
 8222  035650  005737  005522                  TST     INTCHR          ;SEE IF ANY TTY INPUT
 8223  035654  001474                          BEQ     10$             ;BR IF NO INPUT
 8224  035656  105737  003106                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
 8225  035662  001446                          BEQ     12$             ;BR IF YES
 8226  035664  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
 8227  035672  001033                          BNE     4$              ;BR IF NOT (↑C)
 8228  035674  012700  014532                  MOV     #DRVTST,R0      ;SET RETURN ADDR = DRVTST
 8229  035700  105737  003124          2$:     TSTB    XOVLAD          ;SEE IF XXDP CURRENTLY OVERLAID
 8230  035704  001402                          BEQ     6$              ;BR IF NOT
 8231  035706  004737  030536                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP, IF NECESSARY
 8232  035712  000005                  6$:     RESET                   ;RESET ALL DEVICES
 8233  035714  005037  005522                  CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
 8234  035720  005037  001304                  CLR     $TIMES          ;CLEAR THE ITER. COUNT
```

```
                                                ;CLEAR THE XXDP SAVED FLAG
                                                ;RESTORE ERROR HANDLER ADDRESS
                                                ;RESET THE STACK
                                                ;SET RECAL COMMAND
                                                ;DO CLEANUP RECALIBRATE
                                                ;CLEAR THE TEST NO.
                                                ;EXIT FROM TESTS
                                                ;SEE IF (^Z) TYPED
                                                ;BR IF NOT (^Z)
                                                ;SET RETURN ADDR = INPUTP
                                                ;TAKE EXIT
                                                ;SEE IF (^C) TYPED
                                                ;BR IF NOT
                                                ;TYPE "HALT REQUESTED"
                                                ;TYPE "PRESS CONT WHEN RDY"
                                                ;SET HALT ADDRESS
                                                ;TAKE EXIT
                                                ;SEE IF (^G) TYPED
                                                ;BR IF NOT (^G)
                                                ;OPEN SOFTWARE SWR FOR MODIFICATION
                                                ;ENABLE KBD INPUT AGAIN
                                                ;RETURN


;;*****************************************************************
;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO DISK AND PERFORM
;*A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
;*USED.
;;*****************************************************************
036056  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
                                ;LOAD THE R/W BUFFER WITH DATA
036060  012700  064640                  MOV     #RWBUF,R0       ;BUFFER ADDRESS
036064  012701  000400                  MOV     #400,R1         ;400(OCT) WOPDS
036070  010320                  4$:     MOV     R3,(R0)+        ;LOAD A BUFFER WORD
036072  005301                          DEC     R1              ;DECR COUNTER
036074  001375                          BNE     4$              ;BR IF NOT DONE YET
                                ;SET UP PARAMETERS
036076  012765  064640  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
036104  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
                                ;WRITE THE DATA
036112  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
036120  032777  000002  143020          BIT     #BIT1,@$SWR     ;SEE IF WRITES INHIBITED
036126  001002                          BNE     6$              ;BR IF YES
036130  004737  040774                  JSR     PC,DRVCAL       ;WRITE THE DATA
                                ;PERFORM WRITE CHECK
036134  112765  000131  000001  6$:     MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
036142  004737  040774                  JSR     PC,DRVCAL       ;PERFORM WRITE CHECK
036146  104410                          RESREG                  ;RESTORE R0-R5
036150  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.
                                        ;;*****************************************************************
036144  104427                  LDMEM1: SAVREG                          ;SAVE R0-R5
036146  005004                          CLR     R4                      ;R4=0 INDICATES LDMEM1
036150  012703  000001                  MOV     #1,R3                   ;INIT DATA WORD TO 1
036154  000403                          BR      LDM0                    ;PROCEED
036156  104427                  LDMEM2: SAVREG                          ;SAVE R0-R5
036160  012704  177777                  MOV     #-1,R4                  ;R4=177777 INDICATES LDMEM2
036164  013702  005562          LDM0:   MOV     WDSXFR,R2               ;GET NO. OF WORDS
036170  013701  005602                  MOV     PMA,R1                  ;GET MEM ADRS
036174  005737  056246                  TST     $KT11                   ;SEE IF MEM MGT
036200  100012                          BPL     6$                      ;BR IF NOT
036202  042701  160000                  BIC     #160000,R1              ;FORCE RELOCATION THRU PAR6
036206  052701  140000                  BIS     #140000,R1
036212  013737  003176  172354          MOV     SAVPAR,@#KIPAR6         ;INIT PAR6
036220  052737  000001  177572          BIS     #BIT0,@#SR0             ;TURN ON MEM MGT
036226  010321                  6$:     MOV     R3,(R1)+                ;LOAD A WORD INTO BUFFER
036230  005704                          TST     R4                      ;SEE WHICH DATA DESIRED
036232  001001                          BNE     8$                      ;BR IF NOT INCREMENTING DATA
036234  005203                          INC     R3                      ;INCREMENT THE DATA
036236  005737  056246          8$:     TST     $KT11                   ;SEE IF MEM MGT
036242  100010                          BPL     10$                     ;BR IF NOT
036244  032701  020000                  BIT     #BIT13,R1               ;SEE IF OVERFLOW TO NEXT PAGE
036250  001405                          BEQ     10$                     ;BR IF NO OVERFLOW
036252  062737  000200  172354          ADD     #200,@#KIPAR6           ;INCR PAR BY 4K FOR NEW PAGE
036260  042701  020000                  BIC     #BIT13,R1               ;SET PAGE = 6 AGAIN
036264  005302                  10$:    DEC     R2                      ;DECREMENT COUNTER
036266  001357                          BNE     6$                      ;BR IF NOT DONE LOADING YET
036270  005737  056246                  TST     $KT11                   ;SEE IF MEM MGT
036274  100003                          BPL     12$                     ;BR IF NOT
036276  042737  000001  177572          BIC     #BIT0,@#SR0             ;TURN OFF MEM MGT
036304  104410                  12$:    RESREG                          ;RESTORE R0-R5
036306  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.
                                        ;;*****************************************************************
036310  104407                  CKMEM1: SAVREG                          ;SAVE R0-R5
036312  005004                          CLR     R4                      ;R4=0 INDICATES CKMEM1
036314  012703  000001                  MOV     #1,R3                   ;INIT DATA TO 1
036320  000403                          BR      CKM0                    ;PROCEED
036322  104407                  CKMEM2: SAVREG                          ;SAVE R0-R5
036324  012704  177777                  MOV     #-1,R4                  ;R4=177777 INDICATES CKMEM2
036330  013702  005562          CKM0:   MOV     WDSXFR,R2               ;GET NO. OF WORDS
036334  013701  005602                  MOV     PMA,R1                  ;GET MEM ADRS
036340  005037  005532                  CLR     SCRACH
036344  112737  000040  063157          MOVB    #40,DH701+38.           ;RESTORE ERROR MSG PARAMS
```

```
 8347  036356  012737  000007  064532          MOV     #7.DF25+2
 8348  036364  004737  042206                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
 8349  036370  005737  056246                  TST     $KT11           ;SEE IF MEM MGT
 8350  036374  100012                          BPL     6$              ;BR IF NOT
 8351  036376  042701  160000                  BIC     #160000,R1      ;FORCE RELOCATION THRU PAR6
 8352  036402  052701  140000                  BIS     #140000,R1
 8353  036406  013737  003176  172354          MOV     SAVPAR,@#KIPAR6 ;INIT PAR6
 8354  036414  052737  000001  177572          BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
 8355  036422  020321                  6$:     CMP     R3,(R1)+        ;COMPARE A DATA WORD
 8356  036424  001575                          BEQ     16$             ;BR IF DATA COMPARES OK
 8357                                  ;COMPARE ERROR HANDLER
 8358  036426  010237  001202                  MOV     R2,$REG10       ;GET WORD NO.
 8359  036432  163737  005562  001202          SUB     WDSXFR,$REG10
 8360  036440  013737  001202  005570          MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
 8361  036446  004737  034326                  JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
 8362  036452  104407                          SAVREG                  ;SAVE R0-R5
 8363  036454  013700  005564                  MOV     FINCYL,R0       ;GET CYL
 8364  036460  113701  005566                  MOVB    FINTRK,R1       ;GET TRACK
 8365  036464  113702  005567                  MOVB    FINSEC,R2       ;GET SECTOR
 8366  036470  004737  040670                  JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
 8367  036474  104410                          RESREG                  ;RESTORE R0-R5
 8368  036476  032737  001000  005474          BIT     #BADSEC,RECODE
 8369  036504  001145                          BNE     16$             ;BR IF LISTED- DON'T REPORT ERROR
 8370  036506  005737  056246                  TST     $KT11           ;SEE IF MEM MGT
 8371  036512  100406                          BMI     8$              ;BR IF YES
 8372  036514  105037  063157                  CLRB    D4701+38.       ;ADJUST DATA HEADER FOR MSG
 8373  036520  012737  000005  064532          MOV     #5,DF25+2       ;ADJ. ERROR DATA WORD COUNT
 8374  036526  000403                          BR      9$
 8375  036530  013737  172354  001216  8$:     MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
 8376  036536  005737  005532          9$:     TST     SCRACH          ;SEE IF FIRST ERROR IN THIS BLOCK
 8377  036542  001024                          BNE     10$             ;BR IF NOT FIRST ERROR
 8378  036544  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 8379  036550  001411                          BEQ     17$             ;BR IF NOT
 8380  036552  013737  005260  001174          MOV     CRMPHO,$REG5    ;CURRENT UB MAP REG 0
 8381  036560  013737  005256  001176          MOV     CRMPLO,$REG6
 8382  036566  104116                          ERROR   116             ;DATA MISCOMPARE (11/70)
 8383  036570  104117                          ERROR   117
 8384  036572  000401                          BR      11$
 8385  036574  104034                  17$:    ERROP   34              ;TYPE HEADING FOR ERROR MSG
 8386  036576  012737  177777  001174  11$:    MOV     #-1,$REG5       ;INIT CYL NO.
 8387  036604  005037  001176                  CLR     $REG6
 8388  036610  005037  001200                  CLR     $REG7
 8389  036614  005237  005532          10$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
 8390  036620  032777  000001  142316          BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
 8391  036626  001004                          BNE     12$             ;BR TO REPORT ALL ERRORS
 8392  036630  022737  000012  005532          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
 8393  036636  002512                          BLT     21$             ;BR IF ERROR LIMIT EXCEEDED
 8394  036640  023737  001174  005564  12$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
 8395  036646  001010                          BNE     14$             ;BR IF YES
 8396  036650  123737  001176  005566          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
 8397  036656  001004                          BNE     14$             ;BR IF YES
 8398  036660  123737  001200  005567          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
 8399  036666  001412                          BEQ     15$             ;BR IF SAME PACK ADDRESS
 8400  036670  013737  005564  001174  14$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADR FOR PRINTOUT
 8401  036676  113737  005566  001176          MOVB    FINTRK,$REG6
 8402  036704  113737  005567  001200          MOVB    FINSEC,$REG7
```

```
 8403  036706  104115                          ERROR   115             ;TYPE NEW PACK ADRS
 8404  036710  005437  001202          15$:    NEG     $REG10          ;GET WORD NO.
 8405  036714  010337  001204                  MOV     R3,$REG11       ;GOOD DATA
 8406  036720  016137  177776  001206          MOV     -2(R1),$REG12   ;BAD DATA
 8407  036726  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
 8408  036734  005037  001210                  CLR     $REG13
 8409  036740  006137  001212                  ROL     $REG14
 8410  036744  006137  001210                  ROL     $REG13
 8411  036750  063737  005602  001212          ADD     PMA,$REG14
 8412  036756  005537  001210                  ADC     $REG13
 8413  036762  063737  005604  001210          ADD     PMA+2,$REG13
 8414  036770  010137  001214                  MOV     R1,$REG15       ;GET VIRT ADRS FOR PRINTOUT
 8415  036774  162737  000002  001214          SUB     #2,$REG15
 8416  037002  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA
 8417  037004  032777  000100  142126          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
 8418  037012  001022                          BNE     21$             ;BR IF JUST 1
 8419                          ;PROCEED WITH COMPARISONS
 8420  037014  005704                  16$:    TST     R4              ;SEE WHICH DATA DESIRED
 8421  037016  001001                          BNE     18$             ;BR IF NOT INCREASING DATA
 8422  037020  005203                          INC     R3              ;INCREMENT THE DATA WORD
 8423  037022  005737  056246          18$:    TST     $KT11           ;SEE IF MEM MGT
 8424  037026  100010                          BPL     20$             ;BR IF NOT
 8425  037030  032701  020000                  BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
 8426  037034  001405                          BEQ     20$             ;BR IF NO OVERFLOW
 8427  037036  062737  000200  172354          ADD     #200,@#KIPAR6   ;INCR. PAR BY 4K FOR NEW PAGE
 8428  037044  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 8429  037050  005302                  20$:    DEC     R2              ;DECR COUNTER
 8430  037052  001402                          BEQ     21$             ;BR IF DONE COMPARING
 8431  037054  000137  036416                  JMP     6$              ;CONTINUE COMPARING WORDS
 8432  037060  005737  056246          21$:    TST     $KT11           ;SEE IF MEM MGT PRESENT
 8433  037064  100003                          BPL     22$             ;BR IF NOT
 8434  037066  042737  000001  177572          BIC     #BIT0,@#SR0     ;TURN OFF MEM MGT
 8435  037074  104410                  22$:    RESREG                  ;RESTORE R0-R5
 8436  037076  000207                          RTS     PC              ;RETURN
 8437
 8438
 8439
 8440                          ;;*****************************************************************
 8441                          ;*SETUPM - SUBROUTINE TO INIT PARAMS FOR NPR/MEMORY TESTS.
 8442                          ;;*****************************************************************
 8443  037100  013765  005646  000002  SETUPM: MOV     FC,P.CYLN(R5)   ;SET CYL
 8444  037106  113765  005506  000004          MOVB    FS,P.SECT(R5)   ;SET SECTOR
 8445  037114  113765  005652  000005          MOVB    FT,P.TRCK(R5)   ;SET TRACK
 8446  037122  023727  005646  000624          CMP     FC,#624         ;SEE IF CYL SHOULD BE SCALED DOWN
 8447  037130  003403                          BLE     2$              ;BR IF NOT
 8448  037132  012765  000624  000002          MOV     #624,P.CYLN(R5) ;SCALE DOWN THE CYLINDER
 8449  037140  012737  064640  005576  2$:     MOV     #RWBUF,MA       ;GET MEMORY ADDRESS
 8450  037146  005737  000170                  TST     KILLDR          ;SEE IF LOADER CAN BE OVERLAYED
 8451  037152  001003                          BNE     4$              ;BR IF YES
 8452  037154  062737  006000  005576          ADD     #6000,MA        ;LEAVE ROOM TO SAVE LOADER
 8453  037162  042737  003777  005576  4$:     BIC     #3777,MA
 8454  037170  062737  004000  005576          ADD     #4000,MA
 8455  037176  005037  005600                  CLR     MA+2
 8456  037202  000207                          RTS     PC              ;RETURN
 8457
 8458
```

```
 8459
 8460                                   ;;*****************************************************************
 8461                                   ;* REFNEM - THIS SUBROUTINE PERFORMS A SUBSYSTEM COMMAND, WHILE
 8462                                   ;*REPEATEDLY REFERENCING A NON-EXISTENT MEMORY LOCATION (760000). THIS
 8463                                   ;*IS DONE FOR THE PURPOSE OF CAUSING UNIBUS CONTENTION DURING DISK
 8464                                   ;*NPR'S, SINCE THE BUS IS SIEZED FOR 5-20 USEC AT EACH NEM TIME-OUT.
 8465                                   ;*ALL DRIVER PARAMETERS (INCLUDING THE COMMAND) MUST BE SET IN THE
 8466                                   ;*PARAMETER BLOCK ON ENTRY, AND R3 MUST CONTAIN THE TIMING CONSTANT
 8467                                   ;*WHICH REGULATES THE FREQUENCY OF NEM REFERENCES, WHILE WAITING FOR
 8468                                   ;*COMMAND COMPLETION.
 8469                                   ;;*****************************************************************
 8470  037204                           REFNEM:
 8471  037204  004737  035640                  JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
 8472  037210  105037  003113                  CLRB    DONE            ;CLEAR THE DONE FLAG
 8473  037214  004737  041110                  JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
 8474  037220  010537  037236                  MOV     R5,4$           ;SET PARAM BLK ADDRESS
 8475  037224  012737  037272  000004        MOV     #20$,@#ERRVEC   ;SET TIME-OUT VECTOR ADDRESS
 8476  037232  004737  051120                  JSR     PC,C.INIT       ;CALL DRIVER
 8477  037236  000000                   4$:     .WORD
 8478  037240  004737  045470           6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
 8479  037244  105737  003113                  TSTB    DONE            ;DONE SET ?
 8480  037250  001012                          BNE     26$             ;BR IF YES
 8481  037252  010304                          MOV     P3,R4           ;GET COPY OF R3
 8482  037254  001402                          BEQ     12$             ;BR IF TIMER = 0
 8483  037256  005304                   10$:    DEC     R4              ;DECREMENT TIMER
 8484  037260  001376                          BNE     10$             ;BR IF NOT DONE TIMING
 8485  037262  005737  160000           12$:    TST     @#160000        ;REFERENCE NON-EXISTENT MEMORY
 8486  037266  104111                          ERROR   111             ;"NO NEM WHEN EXPECTED"
 8487  037270  000401                          BR      22$
 8488  037272  022626                   20$:    CMP     (SP)+,(SP)+     ;RESTORE THE STACK
 8489  037274  000761                   22$:    BR      6$              ;KEEP WAITING FOR COMMAND COMPLETION
 8490  037276  012737  000006  000004  26$:    MOV     #6,@#ERRVEC     ;RESTORE TIME-OUT VECTOR
 8491  037304  000207                          RTS     PC              ;RETURN
 8492
 8493
 8494                                   ;;*****************************************************************
 8495                                   ;*INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
 8496                                   ;;*****************************************************************
 8497  037306  104407                   INCRMA: SAVREG                  ;SAVE R0-R5
 8498  037310  005001                          CLR     R1
 8499  037312  016500  000012                  MOV     P.WC(R5),R0     ;GET WORD COUNT
 8500  037316  001002                          BNE     4$              ;BR IF NOT 65,536(DEC)
 8501  037320  005201                          INC     R1              ;SET HI BIT
 8502  037322  000401                          BR      6$              ;CONTINUE
 8503  037324  005400                   4$:     NEG     R0              ;GET TRUE WORD COUNT
 8504  037326  000241                   6$:     CLC                     ;DOUBLE THE WORD COUNT TO GET BYTES
 8505  037330  006100                          ROL     R0
 8506  037332  006101                          ROL     R1
 8507  037334  060037  005576                  ADD     R0,MA           ;ADD IT TO COMPUTE NEW MA
 8508  037340  005537  005600                  ADC     MA+2
 8509  037344  060137  005600                  ADD     R1,MA+2
 8510  037350  104410                          RESREG                  ;RESTORE R0-R5
 8511  037352  000207                          RTS     PC              ;RETURN
 8512
 8513
 8514
```

```
                                        ;;*****************************************************************
                                        ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
                                        ;*INTO BSSOFT.
                                        ;;*****************************************************************
037354                                  REDBSF:
037354  004737  030776                          JSR     PC,INITSS       ;INIT THE S.S.
037360  105065  000007                          CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
037364  112765  000121  000001                  MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
037372  012765  000632  000002                  MOV     #632,P.CYLN(R5) ;SET CYL = 632
037400  112765  000002  000005                  MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
037406  012703  000012                          MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
037412  105737  003115                          TSTB    FORMAT
037416  001403                                  BEQ     6$              ;BR IF 22 SECTORS
037420  105265  000004                          INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
037424  005203                                  INC     R3
037426  012765  177400  000012  6$:             MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
037434  012765  004222  000010                  MOV     #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
037442  012737  037616  003036  8$:             MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
037450  105037  003131                          CLRB    WCEFLG          ;INIT THE ERROR FLAG
037454  004737  040774                          JSR     PC,DRVCAL       ;READ THE FACTORY BSF
037460  105737  003131                          TSTB    WCEFLG          ;SEE IF ANY ERRORS
037464  001413                                  BEQ     12$             ;BR IF NOT
037466  062765  000002  000004                  ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
037474  126503  000004                          CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
037500  001360                                  BNE     8$              ;BR IF NOT YET
037502  004737  042206          10$:            JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
037506  104120                                  ERROR   120             ;ABORTING- BAD BSF READ
037510  000137  044476                          JMP     HLTPRG          ;HALT- CAN'T PROCEED
037514  012765  003222  000010  12$:             MOV     #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
037522  110365  000004                          MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
037526  012703  000026                          MOV     #22.,R3         ;SET 22 SECTOR LIMIT
037532  105737  003115                          TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
037536  001402                                  BEQ     14$             ;BR IF YES
037540  012703  000023                          MOV     #19.,R3         ;SET 20 SECTOR LIMIT
037544  012737  037616  003036  14$:             MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
037552  105037  003131                          CLRB    WCEFLG          ;INIT THE ERROR FLAG
037556  004737  040774                          JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
037562  105737  003131                          TSTB    WCEFLG          ;SEE IF ANY ERRORS
037566  001407                                  BEQ     16$             ;BR IF NOT
037570  062765  000002  000004                  ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
037576  126503  000004                          CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
037602  003760                                  BLE     14$             ;BR IF NOT YET
037604  000736                                  BR      10$             ;REPORT ERROR AND ABORT
037606  012737  042466  003036  16$:             MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
037614  000207                                  RTS     PC              ;RETURN

                                        ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
                                        ;*POSSIBLE ERRORS IN READING BAD SECTORS.
037616  032765  130600  000034  BDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
037624  001002                                  BNE     4$              ;BR IF A READ ERROR OCCURRED
037626  000137  042466                          JMP     ERRHDL          ;GO HANDLE OTHER ERROR
037632  105237  003131          4$:             INCB    WCEFLG          ;SET ERROR FLAG
037636  000137  044702                          JMP     RETNML          ;TAKE NORMAL DRIVER RETURN
```

```
                        ;;*****************************************************************
                        ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
                        ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
                        ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
                        ;*2 BSF'S.
                        ;;*****************************************************************
037642  104407          TYPBSF: SAVREG                  ;SAVE R0-R5
037644  005001                  CLR     R1              ;INIT LIMIT COUNTER
037646  005004                  CLR     R4              ;INIT BSF FILE INDICATOR
037650  012700  004232          MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
037654  104401  011224          TYPE    .FACTBS         ;TYPE "FACTORY"
037660  104401  011251          TYPE    .BDSECT         ;TYPE "BAD SECTORS"
037664  104401  062724  3$:     TYPE    .DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
037670  104401  001315          TYPE    .$CRLF
037674  005710  4$:     TST     (R0)            ;SEE IF A SECTOR LISTED HERE
037676  100007          BPL     8$              ;BR IF YES
037700  005701          TST     R1              ;SEE IF FILE IS EMPTY
037702  001032          BNE     14$             ;BR IF NOT EMPTY
037704  104401  011271          TYPE    .NOFALS         ;TYPE "NONE"
037710  104401  001315          TYPE    .$CRLF          ;TYPE <CR>,<LF>
037714  000425          BR      14$
037716  012046  8$:     MOV     (R0)+,-(SP)     ;GET A CYL NO.
037720  104402          TYPOC                   ;TYPE IT
037722  104401  013425          TYPE    .SPACE2         ;TYPE SEPARATORS
037726  011003          MOV     (R0),R3         ;GET TRACK AND SECTOR
037730  105003          CLRB    R3
037732  000303          SWAB    R3              ;GET THE TRACK NO.
037734  010346          MOV     R3,-(SP)
037736  104402          TYPOC                   ;TYPE IT
037740  104401  013425          TYPE    .SPACE2         ;TYPE SEPARATORS
037744  111003          MOVB    (R0),R3         ;GET SECTOR NO.
037746  010346          MOV     R3,-(SP)
037750  104402          TYPOC                   ;TYPE IT
037752  104401  001315          TYPE    .$CRLF
037756  005720          TST     (R0)+           ;INCR THE POINTER
037760  005201          INC     R1              ;INCR THE COUNTER
037762  020127  000176          CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
037766  002742          BLT     4$              ;BR IF NOT YET
037770  005704  14$:     TST     R4              ;SEE IF BOTH FILES TYPED YET
037772  001006          BNE     18$             ;BR IF NOT YET
037774  005204          INC     R4              ;SET INDICATOR FOR SOFT. FILE
037776  012700  003232          MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
040002  104401  011237          TYPE    .SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
040006  000726          BR      3$              ;GO TYPE SOFT. BSF
040010  104410  18$:     RESREG                  ;RESTORE R0-R5
040012  000207          RTS     PC              ;RETURN


                        ;;*****************************************************************
                        ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
                        ;*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
                        ;;*****************************************************************
040014  104407  FINMEM: SAVREG                  ;SAVE R0-R5
040016  016500  000002          MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
040022  116501  000005          MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
```

```
 8627  040026  116502  000004                MOVB    P.SECT(R5),R2   ;SECTOR NO.
 8628  040032  005003                        CLR     R3              ;INIT SECTOR COUNT TO 0
 8629  040034  026500  000030        6$:     CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
 8630  040040  001006                        BNE     8$              ;BR IF NOT EQUAL
 8631  040042  126501  000027                CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
 8632  040046  001003                        BNE     8$              ;BR IF NOT EQUAL
 8633  040050  126502  000026                CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
 8634  040054  001404                        BEQ     12$             ;BR IF ADRS ARE EQUAL
 8635  040056  005203                8$:     INC     R3              ;INCR SECTOR COUNT
 8636  040060  004737  040100                JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
 8637  040064  000763                        BR      6$              ;GO COMPARE ADDRESSES
 8638  040066  000303                12$:    SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
 8639  040070  010337  005562                MOV     R3,WDSXFR       ;STORE IT
 8640  040074  104410                        RESREG                  ;RESTORE R0-R5
 8641  040076  000207                        RTS     PC              ;RETURN
 8642
 8643
 8644
 8645                                ;;*****************************************************************
 8646                                ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
 8647                                ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
 8648                                ;*IN R2.
 8649                                ;;*****************************************************************
 8650  040100  012704  000025        INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
 8651  040104  105737  003115                TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
 8652  040110  001402                        BEQ     4$              ;BR IF YES
 8653  040112  012704  000023                MOV     #19.,R4         ;SET SECTOR LIMIT
 8654  040116  005202                4$:     INC     R2              ;INCR. SECTOR NO.
 8655  040120  020204                        CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
 8656  040122  003407                        BLE     6$              ;BR IF NOT
 8657  040124  005002                        CLR     R2              ;SET SECTOR = 0
 8658  040126  005201                        INC     R1              ;INCR. TRACK NO.
 8659  040130  020127  000002                CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
 8660  040134  003402                        BLE     6$              ;BR IF NOT
 8661  040136  005001                        CLR     R1              ;SET TRACK = 0
 8662  040140  005200                        INC     R0              ;INCR. CYLINDER NO.
 8663  040142  000207                6$:     RTS     PC              ;RETURN
 8664
 8665
 8666
 8667                                ;;*****************************************************************
 8668                                ;*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
 8669                                ;*THIS SUBROUTINE UPDATES PMA,PMA+2,MEM MGT REGS,UNIBUS MAP,P.BALO(R5),
 8670                                ;*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
 8671                                ;*TERMINATED BY A BAD SECTOR ERROR (BSE).
 8672                                ;;*****************************************************************
 8673  040144  104407                MIDXFR: SAVREG                  ;SAVE R0-R5
 8674  040146  004737  040014                JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
 8675  040152  016500  000030                MOV     P.DCYL(R5),R0   ;GET CYL NO.
 8676  040156  116501  000027                MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
 8677  040162  116502  000026                MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
 8678  040166  004737  040100                JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
 8679  040172  010065  000002                MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
 8680  040176  110165  000005                MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
 8681  040202  110265  000004                MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
 8682  040206  013703  005562                MOV     WDSXFR,R3       ;GET NO. OF WORDS XFERRED
```

```
 8683  040212  062703  000400                ADD     #400,R3          ;SKIP BAD SECTOR
 8684  040216  060365  000012                ADD     R3,P.WC(R5)      ;UPDATE P.WC(R5)
 8685  040222  005004                        CLR     R4               ;GET BYTES XFERRED
 8686  040224  006103                        ROL     R3
 8687  040226  006104                        ROL     R4
 8688  040230  060337  005602                ADD     R3,PMA           ;UPDATE PMA,PMA+2
 8689  040234  005537  005604                ADC     PMA+2
 8690  040240  060437  005604                ADD     R4,PMA+2
 8691  040244  013765  005602  000010        MOV     PMA,P.BALO(R5)   ;UPDATE P.BALO(R5)
 8692  040252  013700  005604                MOV     PMA+2,R0
 8693  040256  042700  177774                BIC     #177774,R0
 8694  040262  150065  000007                BISB    R0,P.BAHI(R5)    ;UPDATE P.BAHI(R5)
 8695  040266  005737  056246                TST     $KT11            ;SEE IF MEM MGT
 8696  040272  100002                        BPL     16$
 8697  040274  004737  030334                JSR     PC,PREPAR        ;UPDATE MEM MGT AND UNIBLS MAP
 8698  040300  104410              16$:      RESREG                   ;RESTORE R0-R5
 8699  040302  000207                        RTS     PC               ;RETURN
 8700
 8701
 8702
 8703                                ;;*****************************************************************
 8704                                ;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
 8705                                ;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
 8706                                ;;*****************************************************************
 8707  040304  104407                SVPRMS: SAVREG                   ;SAVE R0-R5
 8708  040306  012700  002622                MOV     #PARM0+2,R0      ;ADRS OF PARAMS
 8709  040312  012701  005550                MOV     #SAVPRS,R1       ;ADRS OF SAVE AREA
 8710  040316  016537  000C12  005562        MOV     P.WC(R5),WDSXFR  ;GET WORD COUNT
 8711  040324  005437  005562                NEG     WDSXFR           ;MAKE IT POSITIVE
 8712  040330  012737  064640  005602        MOV     #RWBUF,PMA       ;INIT PMA TO RWBUF
 8713  040336  005037  005604                CLR     PMA+2            ;INIT PMA+2 TO 0
 8714  040342  005737  005534                TST     PATRN            ;SEE IF DEFAULT DATA TEST
 8715  040346  001414                        BEQ     GTO              ;BR IF YES
 8716  040350  013737  005576  005602        MOV     MA,PMA           ;SET PMA=MA
 8717  040356  013737  005600  005604        MOV     MA+2,PMA+2       ;SET PMA+2=MA+2
 8718  040364  000405                        BR      GTO
 8719  040366  104407                GTPRMS: SAVREG                   ;SAVE R0-R5
 8720  040370  012700  005550                MOV     #SAVPRS,R0       ;ADRS OF SAVE AREA
 8721  040374  012701  002622                MOV     #PARM0+2,R1      ;ADRS OF PARAMS
 8722  040400  012702  000005        GTO:    MOV     #5,R2            ;SET FOR 5 WORDS
 8723  040404  012021                6$:     MOV     (R0)+,(R1)+      ;MOVE A WORD
 8724  040406  005302                        DEC     R2               ;SEE IF DONE YET
 8725  040410  001375                        BNE     6$               ;BR IF NOT YET
 8726  040412  104410                        RESREG                   ;RESTORE R0-R5
 8727  040414  000207                        RTS     PC               ;RETURN
 8728
 8729
 8730
 8731                                ;;*****************************************************************
 8732                                ;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS, IF
 8733                                ;*ENCOUNTERED. THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
 8734                                ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED.
 8735                                ;;*****************************************************************
 8736  040416  010446                TRNSFR: MOV     R4,-(SP)         ;SAVE R4 ON STACK
 8737  040420  005004                        CLR     R4               ;CLEAR BSE ERROR INDICATOR
 8738  040422  105037  003131                CLRB    WCEFLG           ;INIT WCE ERROR FLAG TO 0
```

```
8739  040426  004737  040774          4$:   JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
8740  040432  032737  000100  005474        BIT     #WCERR,RECODE   ;SEE IF WCE ERROR OCCURRED
8741  040440  001403                        BEQ     5$              ;BR IF NOT
8742  040442  112737  000001  003131        MOVB    #1,WCEFLG       ;SET WCE ERROR FLAG
8743  040450  032737  000002  005474  5$:   BIT     #BSERR,RECODE   ;SEE IF BSE ERROR OCCURRED
8744  040456  001406                        BEQ     6$              ;BR IF NOT
8745  040460  005204                        INC     R4              ;INCR BSE ERROR INDICATOR
8746  040462  004737  040144                JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
8747  040466  005765  000012                TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
8748  040472  001355                        BNE     4$              ;BR IF NOT
8749  040474  005704                  6$:   TST     R4              ;SEE IF ANY BSE ERRORS OCCURRED
8750  040476  001411                        BEQ     8$              ;BR IF NOT
8751  040500  004737  040366                JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
8752  040504  004737  040304                JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
8753  040510  005737  056246                TST     $KT11           ;SEE IF MEM MGT PRESENT
8754  040514  100002                        BPL     8$              ;BR IF NOT
8755  040516  004737  030334                JSR     PC,PREPAR       ;PREPARE MEM MGT AND U.M.
8756  040522  012604                  8$:   MOV     (SP)+,R4        ;RESTORE R4
8757  040524  000207                        RTS     PC              ;RETURN
8758
8759
8760
8761                                        ;*****************************************************************
8762                                        .SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
8763                                        ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
8764                                        ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
8765                                        ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
8766                                        ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
8767                                        ;*
8768                                        ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
8769                                        ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
8770                                        ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
8771                                        ;*****************************************************************
8772  040526  010237  001266          SRHTBS: MOV     R2,$TMP2
8773  040532  010337  001270                  MOV     R3,$TMP3
8774  040536  012637  001272                  MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
8775  040542  011402                          MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
8776  040544  012437  001264                  MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
8777  040550  062737  001000  001264          ADD     #1000,$TMP1
8778  040556  005003                          CLR     R3              ;CLEAR R3 FOR COUNT
8779  040560  062702  000010                  ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
8780  040564  005712                  1$:     TST     (R2)            ;TEST IF ALL ONES
8781  040566  100430                          BMI     5$              ;YES-DONE
8782  040570  020012                          CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
8783  040572  001406                          BEQ     3$              ;YES-GO CHECK TRACK
8784  040574  062702  000004                  ADD     #4,R2           ;ELSE BUMP POINTER
8785  040600  020237  001264                  CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
8786  040604  002021                          BGE     5$              ;YES-EXIT
8787  040606  000766                          BR      1$              ;LOOP
8788  040610  005722                  3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
8789  040612  011237  001302                  MOV     (R2),$TMP10     ;GET TRK/SEC WORD
8790  040616  042737  174377  001302          BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
8791  040624  123701  001303                  CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
8792  040630  001402                          BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
8793  040632  005722                          TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
8794  040634  000753                          BR      1$              ;GO TEST NEXT CYL
```

```
8795  040636  005203                  4$:     INC     R3              ;BUMP BAD SECTOR COUNT
8796  040640  016246                          MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
8797  040642  042716  177700                  BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
8798  040646  000746                          BR      1$              ;GO CHECK REST OF FILE
8799  040650  010346                  5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
8800  040652  013702  001266                  MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
8801  040656  013703  001270                  MOV     $TMP3,R3        ;REGISTERS
8802  040662  013746  001272                  MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
8803  040666  000204                          RTS     R4              ;RETURN
8804                                  ;;*****************************************************************
8805                                  .SBTTL  BAD SECTOR CHECK ROUTINE
8806                                  ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
8807                                  ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
8808                                  ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
8809                                  ;*IS NOT LISTED THE FLAG IS RESET.
8810                                  ;;*****************************************************************
8811  040670  104407                  BDSRCH: SAVREG
8812  040672  012737  004222  040704          MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
8813  040700  004437  040526          2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
8814  040704  000000                  1$:     .WORD                   ;TABLE ADDRESS GOES HERE
8815  040706  012603                          MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
8816  040710  001015                          BNE     6$              ;IF ANY, GO TEST WHICH ONES
8817  040712  023727  040704  003222  7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
8818  040720  001404                          BEQ     3$              ;IF YES-EXIT
8819  040722  012737  003222  040704          MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
8820  040730  000763                          BR      2$              ;GO SEARCH IT
8821  040732  042737  001000  005474  3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
8822  040740  104410                  4$:     RESREG
8823  040742  000207                          RTS     PC              ;RETURN
8824  040744  022602                  6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
8825  040746  001403                          BEQ     8$              ;YES-GO SET BIT & EXIT
8826  040750  005303                          DEC     R3              ;DECREMENT BAD SECTOR COUNT
8827  040752  001374                          BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
8828  040754  000756                          BR      7$              ;ELSE GO SEARCH OTHER TABLE
8829  040756  052737  001000  005474  8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
8830  040764  005303                  9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
8831  040766  001764                          BEQ     4$              ;NUMBER
8832  040770  005726                          TST     (SP)+
8833  040772  000774                          BR      9$              ;EXIT
8834
8835
8836
8837                                  ;;*****************************************************************
8838                                  .SBTTL  CALL DRIVER ROUTINE
8839                                  ;*ENTRY JSR PC,DRVCAL
8840                                  ;*              WITH R5 POINTING TO PARAMETER BLOCK
8841                                  ;*RETURN RTS PC
8842                                  ;*
8843                                  ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
8844                                  ;*CALLING THE DRIVEP. THE PARAMETER BLOCK MUST BE SET UP
8845                                  ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
8846                                  ;*BLOCK WHEN THE ROUTINE IS CALLED.
8847                                  ;*
8848                                  ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
8849                                  ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
8850                                  ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
```

```
 8851                                   ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
 8852                                   ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
 8853                                   ;;*****************************************************************
 8854  040774                           DRVCAL:
 8855  040774  004737  034040                   JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
 8856  041000  004737  035640                   JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
 8857  041004  105037  003113                   CLRB    DONE            ;CLEAR DONE FLAG
 8858  041010  105037  003127                   CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
 8859  041014  004737  041110                   JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
 8860  041020  010537  041030                   MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
 8861  041024  004737  051120                   JSR     PC,C.INIT       ;CALL DRIVER
 8862  041030  000000           4$:     .WORD                           ;P.B. ADDRESS GOES HERE
 8863  041032  004737  045470   6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
 8864  041036  105737  003113           TSTB    DONE            ;DONE SET?
 8865  041042  001773                   BEQ     6$              ;NO-LOOP
 8866  041044  105737  003127           TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
 8867  041050  001416                   BEQ     12$             ;BR IF NOT
 8868  041052  105037  003113           CLRB    DONE            ;CLEAR DONE FLAG
 8869  041056  105037  003127           CLRB    DRNAFG          ;CLEAR DRIVE NOT AVAIL FLAG
 8870  041062  010537  041072           MOV     R5,8$           ;GET PARAMETER BLOCK ADDRESS
 8871  041066  004737  051120           JSR     PC,C.INIT       ;RE-ISSUE THE COMMAND
 8872  041072  000000           8$:     .WORD                   ;P.B. ADDRESS GOES HERE
 8873  041074  004737  045470   10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG
 8874  041100  105737  003113           TSTB    DONE            ;DONE SET ?
 8875  041104  001773                   BEQ     10$             ;NO - LOOP
 8876  041106  000207           12$:    RTS     PC              ;YES-RETURN
 8877
 8878
 8879
 8880                                   ;;*****************************************************************
 8881                                   ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
 8882                                   ;;*****************************************************************
 8883  041110  104407           STRCMD: SAVREG                          ;SAVE R0-R5
 8884  041112  012701  005222           MOV     #PRVCMD,R1              ;ADDR OF PREV COMMAND STORAGE
 8885  041116  012700  005236           MOV     #COMSTR,R0              ;ADDR OF CURRENT COMMAND STORAGE
 8886                                   ;STORE PREVIOUS COMMAND
 8887  041122  012021                   MOV     (R0)+,(R1)+
 8888  041124  012021                   MOV     (R0)+,(R1)+
 8889  041126  012021                   MOV     (R0)+,(R1)+
 8890  041130  012021                   MOV     (R0)+,(R1)+
 8891  041132  012021                   MOV     (R0)+,(R1)+
 8892  041134  012021                   MOV     (R0)+,(R1)+
 8893  041136  105737  003134           TSTB    UBMPRS                  ;SEE IF UNIBUS MAP PRESENT
 8894  041142  001414                   BEQ     4$                      ;BR IF NOT
 8895  041144  013737  005260  005254   MOV     CRMPH0,PRMPH0           ;STORE PREV U.B. MAP REG 0
 8896  041152  013737  005256  005252   MOV     CRMPL0,PRMPL0
 8897  041160  013737  170202  005260   MOV     @#MAPH00,CRMPH0 ;STORE CURRENT U.B. MAP REG 0
 8898  041166  013737  170200  005256   MOV     @#MAPL00,CRMPL0
 8899                                   ;STORE CURRENT COMMAND
 8900  041174  012701  005236   4$:     MOV     #COMSTR,R1
 8901  041200  012521                   MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BLK
 8902  041202  012521                   MOV     (R5)+,(R1)+
 8903  041204  012521                   MOV     (R5)+,(R1)+
 8904  041206  012521                   MOV     (R5)+,(R1)+
 8905  041210  012521                   MOV     (R5)+,(R1)+
 8906  041212  012521                   MOV     (R5)+,(R1)+
```

```
8907  041214  104410                          RESREG                  ;RESTORE R0-R5
8908  041216  000207                          RTS     PC              ;RETURN
8909
8910
8911
8912                                  ;;*****************************************************************
8913                                  .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
8914                                  ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
8915                                  ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
8916                                  ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
8917                                  ;*ROUTINE.
8918                                  ;;*****************************************************************
8919  041220  152737  000377  003113  ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
8920  041226  032737  100000  005474          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
8921  041234  001403                          BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
8922  041236  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
8923  041242  000413                          BR      1$              ;EXIT
8924  041244  105737  003116          2$:     TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
8925  041250  001410                          BEQ     1$              ;NO - SKIP TO EXIT
8926  041252  005037  001174                  CLR     $REGS
8927  041256  113737  003116  001174          MOVB    ERRCNT,$REGS    ;GET RETRY COUNT
8928  041264  104101                          ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
8929  041266  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
8930  041272  005037  005474          1$:     CLR     RECODE          ;CLEAR RECOVERY FLAGS
8931  041276  000207                          RTS     PC              ;RETURN
8932                                  ;;*****************************************************************
8933                                  .SBTTL  TYPE ERROR ROUTINE
8934                                  ;*ENTRY -       JSR     PC,TYPERR
8935                                  ;*RETURN -      RTS     PC
8936                                  ;*
8937                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
8938                                  ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
8939                                  ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
8940                                  ;*THE ERROR.
8941                                  ;;*****************************************************************
8942  041300  104407                  TYPERR: SAVREG
8943  041302  105237  003120                  INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
8944  041306  032737  000040  005670          BIT     #BITS.CS        ;SEE IF DRIVE SHOULD BE DROPPED
8945                                                                  ;  IF ERROR LIMIT EXCEEDED
8946  041314  001412                          BEQ     9$              ;BR IF NOT
8947  041316  123727  003120  000024          CMPB    DRVERS,#20.     ;SEE IF 20(DEC) ERRORS EXCEEDED
8948  041324  002406                          BLT     9$              ;BR IF NOT
8949  041326  105037  003120                  CLRB    DRVERS          ;CLEAR DRIVE ERROR COUNT
8950  041332  104401  011103                  TYPE    ,DROPDR         ;TYPE "DROPPING DRIVE"
8951  041336  000137  017470                  JMP     NEWDRV          ;PROCEED TO TEST NEXT DRIVE
8952  041342  032777  020000  137570  9$:     BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
8953  041350  001402                          BEQ     6$              ;BR IF NO
8954  041352  000137  041756                  JMP     20$
8955  041356  005000                  6$:     CLR     R0              ;CLR R0 FOR ERROR NUMBER
8956  041360  005005                          CLR     R5              ;INIT INDENT INDICATOR
8957  041362  005105                          COM     R5
8958  041364  113700  001114                  MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
8959  041370  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
8960  041372  006300                          ASL     R0
8961  041374  006300                          ASL     R0
8962  041376  006300                          ASL     R0
```

```
 8963  041400  062700  001400          1$:   ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
 8964  041404  012037  041424                MOV     (R0)+,2$        ;GET EM POINTER
 8965  041410  001406                       BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 8966  041412  104401  013411                TYPE    ,CR2LF
 8967  041416  104401  056734                TYPE    ,AS2SP2         ;TYPE "** "
 8968  041422  104401                       TYPE                    ;TYPE ERROR MESSAGE (EM)
 8969  041424  000000                2$:    .WORD   0               ;EM POINTER GOES HERE
 8970  041426  012037  041602          3$:   MOV     (R0)+,4$        ;GET DH POINTER
 8971  041432  001467                       BEQ     5$              ;BR IF THERE ISN'T ONE
 8972  041434  104401  001315                TYPE    ,$CRLF
 8973  041440  104401  056672                TYPE    ,TSTMSG         ;TYPE "    TEST "
 8974  041444  013746  001102                MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
 8975  041450  104403                       TYPOS                   ;TYPE IT
 8976  041452     002                       .BYTE   2               ;2 DIGITS
 8977  041453     000                       .BYTE   0               ;SUPPRESS LEADING ZEROS
 8978  041454  104401  013411                TYPE    ,CR2LF
 8979  041460  032777  010000  137452       BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 8980  041466  001133                       BNE     20$             ;BR IF YES
 8981  041470  104401  062026                TYPE    ,DH10S          ;TYPE "PREVIOUS COMMAND :"
 8982  041474  104401  001315                TYPE    ,$CRLF
 8983  041500  104401  062220                TYPE    ,DH10I+10       ;TYPE PREV COMMAND HEADER
 8984  041504  104401  001315                TYPE    ,$CRLF
 8985  041510  012701  000006                MOV     #6,R1           ;SIX COMMAND VALUES
 8986  041514  012702  001236                MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
 8987  041520  012246                30$:   MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
 8988  041522  104402                       TYPOC                   ;TYPE IT
 8989  041524  104401  013425                TYPE    ,SPACE2         ;TYPE SEPARATORS
 8990  041530  005301                       DEC     R1              ;SEE IF 7 VALUES TYPED YET
 8991  041532  001372                       BNE     30$             ;BR IF NOT
 8992  041534  104401  001315                TYPE    ,$CRLF
 8993  041540  104401  013425                TYPE    ,SPACE2         ;INDENT
 8994  041544  104401  062277                TYPE    ,DH10Z          ;TYPE HDR FOR BA DATA
 8995  041550  104401  001315                TYPE    ,$CRLF
 8996  041554  104401  013425                TYPE    ,SPACE2         ;INDENT
 8997  041560  012246                       MOV     (R2)+,-(SP)
 8998  041562  104402                       TYPOC                   ;TYPE PREV. HI BA BITS
 8999  041564  104401  013425                TYPE    ,SPACE2
 9000  041570  011246                       MOV     (R2),-(SP)
 9001  041572  104402                       TYPOC                   ;TYPE PREV. LO BA BITS
 9002  041574  104401  013411                TYPE    ,CR2LF
 9003  041600  104401                       TYPE                    ;TYPE DATA HEADER
 9004  041602  000000                4$:    .WORD   0               ;DH POINTER GOES HERE
 9005  041604  104401  001315                TYPE    ,$CRLF
 9006  041610  005005                       CLR     R5              ;INIT INDENT INDICATOR
 9007  041612  032777  010000  137320  5$:   BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 9008  041620  001056                       BNE     20$             ;BR IF YES
 9009  041622  012001                       MOV     (R0)+,R1        ;GET DT POINTER
 9010  041624  001454                       BEQ     20$             ;BRANCH IF THERE ARE NONE
 9011  041626  012000                       MOV     (R0)+,R0        ;GET DF POINTER
 9012  041630  012002                       MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
 9013  041632  112003                10$:   MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
 9014  041634  105720                       TSTB    (R0)+           ;BUMP PAST FORMAT WORD
 9015  041636  005703                       TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
 9016  041640  001417                       BEQ     14$             ;NO - SKIP DATA PRINT
 9017  041642  005705                       TST     R5              ;SEE IF SHOULD INDENT
 9018  041644  001002                       BNE     11$             ;BR IF NOT
```

```
041646  104401  013425                TYPE    ,SPACE2         ;INDENT
041652  013146                11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
041654  104402                        TYPOC                   ;TYPE IT
041656  005303                        DEC     R3              ;MORE DATA WORDS
041660  001403                        BEQ     12$             ;NO-BRANCH
041662  104401  013425                TYPE    ,SPACE2         ;TYPE SEPARATORS
041666  000771                        BR      11$             ;LOOP
041670  005702                12$:    TST     R2              ;SEE IF <CR>,<LF> NEEDED
041672  001402                        BEQ     14$             ;BR IF NOT
041674  104401  001315                TYPE    ,$CRLF          ;TYPE IT
041700  005302                14$:    DEC     R2              ;MORE DH'S?
041702  003425                        BLE     20$             ;NO-BRANCH
041704  012037  041736        15$:    MOV     (R0)+,16$       ;GET NEXT DH POINTER
041710  105710                        TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
041712  001004                        BNE     34$             ;BR IF YES
041714  104401  001315                TYPE    ,$CRLF          ;SKIP EXTRA LINE
041720  005005                        CLR     R5              ;RE-INIT INDENT INDICATOR
041722  000404                        BR      36$
041724  005105                34$:    COM     R5              ;COMPLEMENT INDENT INDICATOR
041726  001302                        BNE     36$             ;BR IF NO INDENT REQUIRED
041730  104401  013425                TYPE    ,SPACE2         ;INDENT
041734  104401                36$:    TYPE                    ;TYPE DH
041736  000000                16$:    .WORD   0               ;DH POINTER GOES HERE
041740  104401  001315                TYPE    ,$CRLF
041744  105710                        TSTB    (R0)            ;TYPE A DT?
041746  001331                        BNE     10$             ;YES-BRANCH
041750  062700  000002                ADD     #2,R0           ;INCREMENT DF POINTER
041754  000751                        BR      14$             ;SEE IF END OF DF BLOCK
041756  104413                20$:    RESREG
041760  000207                        RTS     PC
                              ;;*************************************************************
                              .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
                              ;*ENTRY:         JSR PC, CONERR
                              ;*RETURN:        RTS PC
                              ;*
                              ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
                              ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
                              ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
                              ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
                              ;*AT THIS TIME.
                              ;;*************************************************************
041762  104407                CONERR: SAVREG                  ;SAVE R0-R5
041764  152737  000377  003113        BISB    #377,DONE       ;SET DONE FLAG
041772  105237  003116                INCB    ERRCNT          ;INCREMENT ERROR COUNT
041776  004737  044712                JSR     PC,TOPROC       ;LOAD RK REGS INTO $REGS
042002  032737  000001  003042        BIT     #BIT0,E.CONT    ;ERROR 0?
042010  001402                        BEQ     1$              ;NO-BRANCH
042012  104064                        ERROR   64              ;CLEAR CONT DID NOT CLEAR ERROR
042014  000470                        BR      7$
042016  032737  000002  003042  1$:   BIT     #BIT1,E.CONT    ;ERROR 1?
042024  001402                        BEQ     2$              ;NO-BRANCH
042026  104065                        ERROR   65              ;NO ATTENTION IN ATTENTION SUM REG
042030  000462                        BR      7$
042032  032737  000004  003042  2$:   BIT     #BIT2,E.CONT    ;ERROR 2?
042040  001407                        BEQ     3$              ;NO-BRANCH
042042  105737  003127                TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
```

```
                                                BEQ     15$             ;BR IF NOT
                                                INCB    REISSU          ;SET FLAG TO RE-ISSUE COMMAND
                                        15$:    ERROR   66              ;UNSOLICITED ATTENTION
                                                BR      7$
                                        3$:     BIT     #BIT3,E.CONT    ;ERROR 3?
                                                BEQ     4$              ;NO-BRANCH
                                                ERROR   67              ;UNEXPECTED DATA TYPE ERROR
                                                BR      7$
                                        4$:     BIT     #BIT4,E.CONT    ;ERROR 4?
                                                BEQ     5$              ;NO-BRANCH
                                                ERROR   70              ;ATTENTION DID NOT RESET WITH CLEAR
                                                BR      7$
                                        5$:     BIT     #BIT5,E.CONT    ;ERROR 5?
                                                BEQ     6$              ;NO-BRANCH
                                                ERROR   71              ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
                                                BR      7$
                                        6$:     BIT     #BIT8,E.CONT
                                                BEQ     8$
                                                ERROR   72              ;DATA LATE WHEN UNLOADING HEADER
                                        8$:     BIT     #BIT9,E.CONT
                                                BEQ     9$
                                                ERROR   73              ;CONTROLLER ERROR DURING DRIVER SERVICE
                                        9$:     BIT     #BIT10,E.CONT
                                                BEQ     10$
                                                ERROR   74              ;DRIVE DETECTED PARITY ERROR
                                        10$:    BIT     #BIT15,E.CONT
                                                BEQ     11$
                                                ERROR   52              ;MULTIPLE DRIVE SELECT
                                        11$:    ERROR   75              ;UNDEFINED ERROR
                                        7$:     CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
                                                JMP     BGNRTY          ;GO DO RETRY
                                        ;;*****************************************************************
                                        .SBTTL  REPORT SUPPORT ROUTINE
                                        ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
                                        ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING.  ALL THESE MAY
                                        ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
042206                                  REPSUP:
042206  104407                                  SAVREG
042210  005037  005476                          CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
042214  116537  000001  005476                  MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
042222  012700  001162                          MOV     #$REG0,R0       ;FOR REPORTING
042226  016520  000002                          MOV     P.CYLN(R5),(R0)+
042232  116520  000005                          MOVB    P.TRCK(R5),(R0)+
042236  105020                                  CLRB    (R0)+
042240  116520  000004                          MOVB    P.SECT(R5),(R0)+
042244  105020                                  CLRB    (R0)+
042246  016520  000012                          MOV     P.WC(R5),(R0)+
042252  012700  001174                          MOV     #$REG5,R0
042256  116503  000007                          MOVB    P.BAHI(R5),R3
042262  042703  177774                          BIC     #177774,R3
042266  010337  001256                          MOV     R3,$REG36       ;HI BA BITS
042272  016537  000010  001260                  MOV     P.BALO(R5),$REG37 ;LO BA BITS
042300  016520  000016                          MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
042304  016520  000020                          MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
042310  016520  000030                          MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
042314  016520  000026                          MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
```

```
9131  042320  016520  000022                MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
9132                                                                  ;FOR ALL REPORTS (TO BE
9133  042324  016520  000024                MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
9134  042330  016520  000032                MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
9135  042334  016520  000036                MOV     P.DS(R5),(R0)+
9136  042340  016520  000034                MOV     P.ER(R5),(R0)+
9137  042344  016520  000040                MOV     P.A00(R5),(R0)+
9138  042350  016520  000042                MOV     P.B00(R5),(R0)+
9139  042354  016520  000044                MOV     P.A01(R5),(R0)+
9140  042360  016520  000046                MOV     P.B01(R5),(R0)+
9141  042364  016520  000050                MOV     P.A10(R5),(R0)+
9142  042370  016520  000052                MOV     P.B10(R5),(R0)+
9143  042374  016520  000054                MOV     P.A11(R5),(R0)+
9144  042400  016520  000056                MOV     P.B11(R5),(R0)+
9145                              ;STORE PREVIOUS COMMAND FOR PRINTOUT
9146  042404  012701  001236                MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
9147  042410  012700  005222                MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
9148  042414  112021                        MOVB    (R0)+,(R1)+     ;DRIVE NO.
9149  042416  105021                        CLRB    (R1)+
9150  042420  112021                        MOVB    (R0)+,(R1)+     ;COMMAND
9151  042422  105021                        CLRB    (R1)+
9152  042424  012021                        MOV     (R0)+,(R1)+     ;CYL ADDRESS
9153  042426  116021  000001                MOVB    1(R0),(R1)+     ;TRACK
9154  042432  105021                        CLRB    (R1)+
9155  042434  111021                        MOVB    (R0),(R1)+      ;SECTOR
9156  042436  105021                        CLRB    (R1)+
9157  042440  016021  000006                MOV     6(R0),(R1)+     ;WORD COUNT
9158  042444  116003  000003                MOVB    3(R0),R3        ;HI BA BITS
9159  042450  042703  177774                BIC     #177774,R3
9160  042454  010321                        MOV     R3,(R1)+
9161  042456  016011  000004                MOV     4(R0),(R1)      ;LO BA BITS
9162  042462  104410                        RESREG
9163  042464  000207                        RTS     PC
9164                          ;;**************************************************************
9165                          .SBTTL  REPORT ERROR ROUTINE
9166                          ;* ENTRY         JSR     PC, ERRHDL
9167                          ;*RETURN        RTS     PC
9168                          ;*
9169                          ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
9170                          ;*IN THE OPERATION.  THE ROUTINE DETERMINES WHICH COMMAND WAS
9171                          ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
9172                          ;;**************************************************************
9173
9174  042466  104407                ERRHDL: SAVREG
9175  042470  152737  000377  003113        BISB    #377,DONE       ;SET DONE FLAG
9176  042476  105237  003116                INCB    ERRCNT          ;INCREMENT ERROR COUNT
9177  042502  005037  005474                CLR     RECODE          ;CLEAR RECOVERY CODE WORD
9178  042506  032737  000400  005474 ER2ENT: BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
9179  042514  001402                        BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
9180  042516  012705  002704                MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
9181  042522  012737  044664  003036 52$:    MOV     #RETNL,A.ABNL   ;SET NOW ABNORMAL AND NORMAL RETURN FOR
9182  042530  012737  044702  003034        MOV     #RETNM,A.NORM   ;DRIVER OPERATIONS IN ERROR PROCESSING
9183  042536  004737  042206                JSR     PC,REPSUP       ;GO SET UP REGISTERS FOR REPORT
9184                                                                  ;NOW BEGIN TESTING THE ERROR
9185                                                                  ;BITS. THE SEQUENCE IN
9186                                                                  ;WHICH THEY ARE TESTED IS
```

```
                                                                        ;CONSIDERED SIGNIFICANT IN
                                                                        ;THAT ERRORS OF A MORE
                                                                        ;CATASTROPHIC NATURE ARE FIRST
                                                                        ;TESTED.
                                                                        ;
                                                                        ;IF AN ERROR IS FOUND SET,
                                                                        ;THAT ERROR IS REPORTED AND
                                                                        ;THE REPORTING IS TERMINATED.
                                                                        ;IF ADDITIONAL ERRORS ARE SET,
                                                                        ;THE RK611 REGISTER PRINTOUTS
                                                                        ;WILL SHOW THIS BUT THE
                                                                        ;REGISTER CONTENTS MUST BE
                                                                        ;MANUALLY DECODED TO LOCATE THE
                                                                        ;SECOND ERROR
042542  016504  000034                  MOV     P.ER(R5),R4             ;SET R4 TO ERROR REGISTER
042546  032765  020000  000020          BIT     #UPE,P.CS2(R5)          ;TEST UPE. IF UES-SET
042554  001406                          BEQ     1$                      ;REPORT ERROR
042556  104001                          ERROR   1
042560  052737  000200  005474          BIS     #ABORT,RECODE
042566  000137  044236                  JMP     37$
042572  032765  004000  000020  1$:     BIT     #NEM,P.CS2(R5)          ;TEST NON-EXISTANT MEMORY
042600  001406                          BEQ     2$
042602  104002                          ERROR   2
042604  052737  000200  005474          BIS     #ABORT,RECODE
042612  000137  044236                  JMP     37$
042616  032765  010000  000020  2$:     BIT     #NED,P.CS2(R5)          ;TEST NON-EXISTANT DRIVE
042624  001412                          BEQ     3$
042626  032765  000400  000020          BIT     #UFE,P.CS2(R5)          ;TEST IF NED & UFE BOTH SET
042634  001403                          BEQ     38$
042636  104035                          ERROR   35                      ;NED & UFE BOTH SET ERROR
042640  000137  044236                  JMP     37$
042644  104003                  38$:     ERROR   3                       ;NED ONLY
042646  000137  044236                  JMP     37$
042652  032765  000400  000020  3$:     BIT     #UFE,P.CS2(R5)          ;TEST UNIT FIELD ERROR
042660  001412                          BEQ     4$
042662  032765  010000  000020          BIT     #NED,P.CS2(R5)          ;TEST IF UFE & NED BOTH SET
042670  001403                          BEQ     39$                     ;NO-SKIP
042672  104035                          ERROR   35                      ;REPORT NED & UFE BOTH SET
042674  000137  044236                  JMP     37$
042700  104004                  39$:     ERROR   4                       ;REPORT UFE ONLY
042702  000137  044236                  JMP     37$
042706  032765  000100  000014  4$:     BIT     #CMDTO,P.PRST(R5) ;
042714  001423                          BEQ     5$
042716  004737  044712                  JSR     PC,TOPROC               ;GO PROCESS TIMEOUT
042722  032737  002000  005474          BIT     #TWOTOS,RECODE          ;2ND TIMEOUT IN TIMEOUT PROC^
042730  001007                          BNE     40$                     ;YES - SKIP TO ERROR 56
042732  032737  000400  005474          BIT     #LEV2ER,RECODE          ;TEST IF LEVEL 2 ERROR
042740  001006                          BNE     41$                     ;YES - SKIP TO ERROR 57
042742  104005                          ERROR   5                       ;ELSE MAKE FULL TIMEOUT REPORT
042744  000137  044236                  JMP     37$
042750  104056                  40$:     ERROR   56                      ;TWO TIMEOUTS ERROR REPORT
042752  000137  044236                  JMP     37$
042756  104057                  41$:     ERROR   57                      ;2ND LEVEL ERROR REPORT
042760  000137  042506                  JMP     ER2ENT                  ;GO BUILD AND MAKE 2ND REPORT
042764  032765  010000  000014  5$:     BIT     #DRVSZD,P.PRST(R5) ;SEE IF DRIVE SIEZED BY OTHER PORT
042772  001431                          BEQ     65$                     ;BR IF DRIVE IS AVAILABLE
```

```
 9243  043274  105737  003126                TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
 9244  043300  001003                        BNE     63$             ;BR IF DUAL-ACCESS TEST
 9245  043302  104107                        ERROR   107             ;DRIVE SIEZED BY OTHER PORT
 9246  043304  000137  044236                JMP     37$
 9247  043310  105237  003127        63$:    INCB    DRNAFG          ;SET DRIVE NOT AVAILABLE FLAG
 9248  043314  012762  100000  000000        MOV     #100000,RKCS1(R2) ;CLEAR CONTROLLER ERROR
 9249  043322  013737  003056  003104        MOV     W.MIN,W.DRV     ;SET TIMER FOR ABOUT A MINUTE
 9250  043330  113700  005500                MOVB    DRIVE,R0        ;DRIVE NUMBER
 9251  043334  116037  003072  003071        MOVB    I.DRV(R0),INTMSK ;SET MASK FOR THIS DRIVE
 9252  043342  113737  003071  003070        MOVB    INTMSK,W.TIME   ;SET DRIVE NUMBER FOR THIS DRIVE
 9253  043350  105037  003113                CLRB    DONE            ;CLEAR THE DONE FLAG
 9254  043354  000207                        RTS     PC              ;GO WAIT FOR DRIVE ATT'N
 9255  043356  032765  020000  000016  65$:   BIT     #SPAR,P.CS1(R5) ;TEST D TO C PARITY ERROR
 9256  043364  001406                        BEQ     6$
 9257  043366  104006                        ERROR   6
 9259  043370  052737  004000  005474        BIS     #RCLREQ,RECODE
 9259  043376  000137  044236                JMP     37$
 9260  043102  032704  000010          6$:   BIT     #DRPAR,R4       ;TEST DRIVE DETECTED PARITY ERROR
 9261  043106  001406                        BEQ     7$
 9262  043110  104007                        ERROR   7
 9263  043112  052737  004000  005474        BIS     #RCLREQ,RECODE
 9264  043120  000137  044236                JMP     37$
 9265  043124  032765  000010  000036  7$:   BIT     #ACLO,P.DS(R5)  ;TEST AC LOW
 9266  043132  001403                        BEQ     8$
 9267  043134  104010                        ERROR   10
 9268  043136  000137  044236                JMP     37$
 9269  043142  032765  000020  000036  8$:   BIT     #SPDLSS,P.DS(R5) ;TEST SPEED LOSS
 9270  043150  001403                        BEQ     24$
 9271  043152  104011                        ERROR   11
 9272  043154  000137  044236                JMP     37$
 9273  043160  032765  004000  000016  24$:   BIT     #CTO,P.CS1(R5)  ;TEST FOR CONTROLLER TIMEOUT
 9274  043166  001401                        BEQ     25$
 9275  043170  104027                        ERROR   27
 9276  043172  032704  000001          25$:   BIT     #ILC,R4         ;TEST ILLEGAL FUNCTION CODE
 9277  043176  001403                        BEQ     10$
 9278  043200  104012                        ERROR   12
 9279  043202  000137  044236                JMP     37$
 9280  043206  032765  002000  000020  10$:   BIT     #PGE,P.CS2(R5)  ;TEST PROGRAMMING ERROR
 9281  043214  001403                        BEQ     11$
 9282  043216  104013                        ERROR   13
 9283  043220  000137  044236                JMP     37$
 9284  043224  032704  000004          11$:   BIT     #ILF,R4         ;TEST ILLEGAL DRIVE FUNCTION
 9285  043230  001403                        BEQ     12$
 9286  043232  104014                        ERROR   14
 9287  043234  000137  044236                JMP     37$
 9288  043240  032704  000040          12$:   BIT     #DTYE,R4        ;TEST DRIVE TYPE ERROR
 9289  043244  001403                        BEQ     13$
 9290  043246  104015                        ERROR   15
 9291  043250  000137  044236                JMP     37$
 9292  043254  032704  000020          13$:   BIT     #FMTE,R4        ;TEST FORMAT ERROR
 9293  043260  001403                        BEQ     14$
 9294  043262  104016                        ERROR   16
 9295  043264  000137  044236                JMP     37$
 9296  043270  032704  004000          14$:   BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
 9297  043274  001403                        BEQ     15$
 9298  043276  104017                        ERROR   17
```

```
 9299  043300  000137  044236                    JMP     37$
 9300  043304  032704  040000          15$:      BIT     #UNS,R4         ;TEST DRIVE UNSAFE
 9301  043310  001406                            BEQ     16$
 9302  043312  104020                            ERROR   20
 9303  043314  052737  000200  005474            BIS     #ABORT,RECODE
 9304  043322  000137  044336                    JMP     ALLTRM
 9305  043326  032704  000002          16$:      BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
 9306  043332  001406                            BEQ     17$
 9307  043334  104021                            ERROR   21
 9308  043336  052737  004000  005474            BIS     #RCLREQ,RECODE
 9309  043344  000137  044236                    JMP     37$
 9310  043350  032704  001000          17$:      BIT     #COE,R4         ;TEST CYLINDER OVERFLOW
 9311  043354  001406                            BEQ     18$
 9312  043356  104022                            ERROR   22
 9313  043360  052737  004000  005474            BIS     #RCLREQ,RECODE
 9314  043366  000137  044236                    JMP     37$
 9315  043372  032704  002000          18$:      BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
 9316  043376  001406                            BEQ     19$
 9317  043400  104023                            ERROR   23
 9318  043402  052737  004000  005474            BIS     #RCLREQ,RECODE
 9319  043410  000137  044236                    JMP     37$
 9320  043414  032765  000040  000036  19$:      BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
 9321  043422  001406                            BEQ     20$
 9322  043424  104024                            ERROR   24
 9323  043426  052737  004000  005474            BIS     #RCLREQ,RECODE
 9324  043434  000137  044236                    JMP     37$
 9325  043440  032704  010000          20$:      BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
 9326  043444  001406                            BEQ     21$
 9327  043446  104025                            ERROR   25
 9328  043450  052737  000200  005474            BIS     #ABORT,RECODE
 9329  043456  000137  044236                    JMP     37$
 9330  043462  032765  100000  000020  21$:      BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
 9331  043470  001403                            BEQ     22$
 9332  043472  104026                            ERROR   26
 9333  043474  000137  044236                    JMP     37$
 9334  043500  032704  020000          22$:      BIT     #OPI,R4         ;TEST IF OPI ERROR
 9335  043504  001457                            BEQ     29$
 9336  043506  052737  000010  005474            BIS     #OPIERR,RECODE
 9337  043514  105737  003111                    TSTB    DERCNT          ;TEST IF FIRST ERROR
 9338  043520  001402                            BEQ     50$
 9339  043522  000137  044236                    JMP     37$             ;NO - SKIP REPORT
 9340  043526  032737  000400  005474  50$:      BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
 9341  043534  001403                            BEQ     26$             ;HAS ALREADY OCCURRED
 9342  043536  104054                  27$:      ERROR   54
 9343  043540  000137  044236                    JMP     37$             ;GET OUT OF ERROR REPORT
 9344  043544  004737  045342          26$:      JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
 9345  043550  004737  045062                    JSR     PC,RDHDO        ;GET HEADER OF SECTOR 0
 9346  043554  032737  000400  005474            BIT     #LEV2ER,RECODE  ;TEST IF ERROR GETTING HDR
 9347  043562  001025                            BNE     28$             ;IF YES-GO MAKE ABREVIATED REPORT
 9348  043564  013702  003026                    MOV     RKBAS,R2        ;STORE HEADER 0 INTO REGISTERS
 9349  043570  042762  000100  000000            BIC     #IE,RKCS1(R2)   ;RESET INTERRUPT ENABLE
 9350  043576  016237  000024  001202            MOV     RKDB(R2),$REG10 ;FOR REPORTING
 9351  043604  016237  000024  001204            MOV     RKDB(R2),$REG11
 9352  043612  016237  000024  001206            MOV     RKDB(R2),$REG12
 9353  043620  032762  100000  000000            BIT     #CERR,RKCS1(R2) ;TEST IF ERROR DURING STORAGE
 9354  043626  001343                            BNE     27$
```

```
 9355  043630  104030                        ERROR   30              ;MAKE OPI REPORT
 9356  043632  000137  044236                JMP     37$
 9357  043636  104054                28$:    ERROR   54
 9358  043640  000137  042506                JMP     ER2ENT          ;GO MAKE 2ND LEVEL REPORT
 9359  043644  032704  000400        29$:    BIT     #HVRC,R4        ;TEST IF HVRC ERROR
 9360  043650  001457                        BEQ     23$
 9361  043652  052737  000004  005474        BIS     #HVRCER,RECODE
 9362  043660  105737  003111                TSTB    DERCNT          ;TEST IF FIRST ERROR
 9363  043664  001402                        BEQ     30$             ;YES - REPORT
 9364  043666  000137  044236                JMP     37$             ;JUMP TO RETURN
 9365  043672  032737  000400  005474  30$:  BIT     #LEV2ER,RECODE  ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
 9366  043700  001403                        BEQ     31$             ;OCCURRED. NO-SKIP EXIT
 9367  043702  104055                51$:    ERROR   55
 9368  043704  000137  044236                JMP     37$             ;GET OUT OF ERROR REPORT
 9369  043710  004737  045342        31$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
 9370  043714  004737  045062                JSR     PC,RDHDO        ;GO GET HDR 0
 9371  043720  032737  000400  005474        BIT     #LEV2ER,RECODE  ;TEST IF ERROR IN GETTING HDR
 9372  043726  001025                        BNE     32$             ;IF YES-GO MAKE ABREVIATED REPORT
 9373  043730  013702  003026                MOV     RKBAS,R2        ;GET RK611 BASE ADDRESS
 9374  043734  042762  000100  000000        BIC     #IE,RKCS1(R2)   ;CLEAR INTERRUPT ENABLE
 9375  043742  016237  000024  001202        MOV     RKDB(R2),$REG10 ;STORE OFF HEADER
 9376  043750  016237  000024  001204        MOV     RKDB(R2),$REG11
 9377  043756  016237  000024  001206        MOV     RKDB(R2),$REG12
 9378  043764  032762  100000  000000        BIT     #CERR,RKCS1(R2) ;TEST IN ANY ERROR IN UNLOAD
 9379  043772  001343                        BNE     51$             ;IY YES - GO MAKE SHORT REPORT
 9380  043774  104031                        ERROR   31              ;MAKE FULL REPORT
 9381  043776  000137  044236                JMP     37$
 9382  044002  104055                32$:    ERROR   55              ;ABREVIATED HVRC ERROR RPORT
 9383  044004  000137  042506                JMP     ER2ENT          ;GO REPORT 2ND LEVEL ERROR
 9384  044010  032704  000200        23$:    BIT     #BSE,R4         ;TEST FOR BAD SECTOR ERROR
 9385  044014  001430                        BEQ     33$             ;NO - SKIP
 9386  044016  052737  000002  005474        BIS     #BSERR,RECODE   ;SET ERROR FLAG
 9387  044024  016500  000030                MOV     P.DCYL(R5),R0   ;GET CYL NO.
 9388  044030  116501  000027                MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
 9389  044034  042701  177774                BIC     #177774,R1      ;CLEAR ALL BITS EXCEPT TRACK
 9390  044040  016502  000026                MOV     P.DTS(R5),R2    ;GET SECTOR IN ERROR
 9391  044044  042702  177740                BIC     #177740,R2      ;CLEAR ALL BITS EXCEPT SECTOR
 9392  044050  004737  040670                JSR     PC,BDSRCK       ;GO SEE IF THIS SECTOR LISTED
 9393  044054  032737  001000  005474        BIT     #BADSEC,RECODE  ;TEST RESULT
 9394  044062  001065                        BNE     37$             ;YES - EXIT, NO ERROR OR REPORT
 9395  044064  104104                        ERROR   104             ;ELSE REPORT BAD BSE
 9396  044066  042737  000002  005474        BIC     #BSERR,RECODE   ;RESET BSE ERROR FLAG
 9397  044074  000460                        BR      37$             ;GO EXIT
 9398  044076  032704  100000        33$:    BIT     #DCK,R4         ;TEST IF DATA CHECK
 9399  044102  001435                        BEQ     36$
 9400  044104  052737  000020  005474        BIS     #DCKERR,RECODE  ;SET DATA CHECK ERROR IN RECOVERY CODE
 9401  044112  032704  000100                BIT     #ECH,R4         ;TEST IF ECC IS HARD. IF
 9402  044116  001406                        BEQ     34$             ;YES SET UNCORRECTABLE IN
 9403  044120  052737  000040  005474        BIS     #ECCNC,RECODE   ;RECOVERY FLAG AND A 0 IN
 9404  044126  005037  001206                CLR     $REG12          ;REG12 TO INDICATE UNCORRECTABLE,
 9405  044132  000403                        BR      35$             ;A 1 IN REG 12 FOR CORRECTABLE
 9406  044134  012737  000001  001206  34$:  MOV     #1,$REG12
 9407  044142  105737  003111        35$:    TSTB    DERCNT          ;TEST IF FIRST ERROR
 9408  044146  001033                        BNE     37$             ;NO SKIP REPORT
 9409  044150  004737  045216                JSR     PC,GTPKAD       ;GO GET PACK ADDRESS OF ERROR
 9410  044154  016537  000060  001202        MOV     P.EPOS(R5),$REG10 ;STORE ECC POSITION &
```

```
9411  044162  016537  000062  001204          MOV     P.EPAT(R5),$REG11 ;PATTERN
9412  044170  104032                          ERROR   32                ;REPORT DCK ERROR
9413  044172  000137  044236                  JMP     37$
9414  044176  032765  040000  000020  36$:    BIT     #WCE,P.CS2(R5)    ;TEST WRITE CHECK ERROR
9415  044204  001414                          BEQ     37$
9416  044206  042737  000200  005474          BIC     #ABORT,RECODE     ;CLEAR ABORT & SET WRITE
9417  044214  052737  000100  005474          BIS     #WCERR,RECODE     ;CHECK ERROR IN RECODE
9418  044222  105737  003111                  TSTB    DERCNT            ;TEST IF FIRST ERROR
9419  044226  001003                          BNE     37$               ;NO - SKIP
9420  044230  004737  045216                  JSR     PC,GTPKAD         ;GO GET ADDRESS OF ERROR
9421  044234  104033                          ERROR   33                ;REPORT WCE
9422
9423  044236  032765  000020  000014  37$:    BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
9424  044244  001404                          BEQ     43$
9425  044246  104036                          ERROR   36
9426  044250  052737  000200  005474          BIS     #ABORT,RECODE
9427
9428  044256  032765  000040  000014  43$:    BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
9429  044264  001404                          BEQ     44$
9430  044266  104037                          ERROR   37
9431  044270  052737  000200  005474          BIS     #ABORT,RECODE
9432
9433  044276  032765  004000  000014  44$:    BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
9434  044304  001404                          BEQ     46$
9435  044306  104040                          ERROR   40
9436  044310  052737  000200  005474          BIS     #ABORT,RECODE
9437
9438  044316  032765  000010  000014  46$:    BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
9439  044324  001404                          BEQ     ALLTRM
9440  044326  104042                          ERROR   42
9441  044330  052737  000200  005474          BIS     #ABORT,RECODE
9442
9443
9444                                  ;ALL ERRORS MUST EXIT THROUGH THIS POINT
9445
9446  044336  012705  002620          ALLTRM: MOV     #PARM0,R5         ;RESTORE PARAMETER BLOCK SELECTION
9447  044342  012737  042466  003036          MOV     #ERRHDL,A.ABNL    ;RESTORE INTERRUPT VECTORS FOR RETRY
9448  044350  012737  041220  003034          MOV     #ERRFRE,A.NORM    ;DRIVER OPERATIONS, IF ANY
9449  044356  032737  000200  005474          BIT     #ABORT,RECODE     ;IF ABORT IS NOT SET AND
9450  044364  001043                          BNE     48$               ;THE DRIVE IS READY (HAS NOT
9451                                                                    ;CYCLED DOWN)
9452  044366  013702  003026                  MOV     RKBAS,R2          ;GET BASE ADDRESS
9453  044372  032762  000200  000012          BIT     #RDY,RKDS(R2)     ;TEST IF DRIVE READY SET
9454  044400  001004                          BNE     47$               ;RECALIBRATE REQUIRED BIT IS SET
9455  044402  052737  000200  005474          BIS     #ABORT,RECODE     ;ELSE ABORT WITH ABORT MESSAGE
9456  044410  000431                          BR      46$
9457  044412  032737  004000  005474  47$:    BIT     #RCLREQ,RECODE    ;IF RECALIBRATE IS REQUIRED
9458  044420  001434                          BEQ     BGNRTY            ;FOR RETRY SET UP PARAM
9459  044422  112737  000113  000001          MOVB    #RECAL,P.CMND     ;BLOCK TO DO IT.
9460  044430  012737  044664  003036          MOV     #RETANL,A.ABNL
9461  044436  004737  040774                  JSR     PC,DRVCAL
9462  044442  012737  042466  003036          MOV     #ERRHDL,A.ABNL    ;RESTORE ERROR RETURN
9463  044450  032737  000400  005474          BIT     #LEV2ER,RECODE    ;IF AN ERROR OCCURRED IN THE
9464  044456  001415                          BEQ     BGNRTY            ;RECAL ATTEMP SET ABORT,
9465  044460  052737  000200  005474          BIS     #ABORT,RECODE     ;PRINT THE RECAL ERROR MESSAGE, AND
9466  044466  104060                          ERROR   60                ;GO REPORT DETAILS
```

```
9467  044470  000137  042506                    JMP     ER2ENT
9468  044474  104061                    48$:    ERROR   61              ;REPORT ABORT-RETRY FAILED
9469
9470                                    ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
9471                                    ;IF THE DRIVE READY BIT IS RESET.
9472
9473  044476  000000                    HLTPRG: HALT
9474  044500  105037  003116                    CLRB    ERRCNT          ;CLEAR ERROR COUNT
9475  044504  000005                            RESET                   ;RESET ALL DEVICES
9476  044506  000137  013522                    JMP     CMSTRT
9477
9478
9479                                    ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
9480                                    ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
9481                                    ; THE INITIATING ROUTINE FOR RETRY.  ANY OTHER ERROR IS TO BE
9482                                    ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
9483                                    ;FLAG IS SET AND PROGRAM HALTS.
9484
9485  044512  032737  000136  005474  BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
9486  044520  001404                            BEQ     3$
9487  044522  052737  100000  005474  9$:     BIS     #ANYDER,RECODE
9488  044530  000453                            BR      2$
9489  044532                          3$:
9490  044532  105737  003133                    TSTB    NORTRY          ;SEE IF "NO-RETRY" FLAG SET
9491  044536  001371                            BNE     9$              ;BR IF YES
9492  044540  032737  001000  005474            BIT     #BADSEC,RECODE  ;TEST IF BAD SECTOR FLAG SET
9493  044546  001044                            BNE     2$              ;IF YES-EXIT TO CALLER
9494  044550  123737  003116  003117            CMPB    ERRCNT,ERRLMT   ;BEGIN RETRY IF ERROR COUNT HAS
9495  044556  001012                            BNE     1$              ;NOT BEEN EXCEEDED
9496  044560  005037  001174                    CLR     $REGS
9497  044564  113737  003116  001174            MOVB    ERRCNT,$RFGS    ;GET ERROR RETRY COUNT
9498  044572  104102                            ERROR   102             ;REPORT RETRY UNSUCCESSFUL
9499  044574  052737  000200  005474            BIS     #ABORT,RECODE   ;SET ABORT & QUIT
9500  044602  000735                            BR      HLTPRG
9501  044604  013702  003026            1$:     MOV     RKBAS,R2        ;GET RK BASE ADDRESS
9502  044610  112765  000177  000001            MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
9503  044616  004737  040774                    JSR     PC,DRVCAL       ;GO DO IT
9504  044622  012700  005236                    MOV     #COMSTR,R0      ;GO AND REESTABLISH THE COMMAND
9505  044626  012025                            MOV     (R0)+,(R5)+     ;AS IT WAS ENTERED INTO THE
9506  044630  012025                            MOV     (R0)+,(R5)+     ;PARAMETER BLOCK
9507  044632  012025                            MOV     (R0)+,(R5)+
9508  044634  012025                            MOV     (R0)+,(R5)+
9509  044636  012025                            MOV     (R0)+,(R5)+
9510  044640  012025                            MOV     (R0)+,(R5)+
9511  044642  012705  002620                    MOV     #PARM0,R5
9512  044646  012737  000000  177776            MOV     #PR0,@#PSW      ;LOWER PRIORITY TO ALLOW INTERRUPT
9513  044654  004737  040774                    JSR     PC,DRVCAL       ;CALL DRIVER
9514  044660  104410                    2$:     RESREG                  ;IF RETURN GETS HERE, NO ERROR
9515  044662  000207                            RTS     PC              ;OCCURRED, RECOVERY WAS SUCCESSFUL
9516
9517
9518  044664  152737  000377  003113  RETANL: BISB    #377,DONE       ;SET DONE
9519  044672  052737  000400  005474            BIS     #LEV2ER,RECODE  ;SET LEVEL TWO ERROR
9520  044700  000207                            RTS     PC
9521  044702  152737  000377  003113  RETNML: BISB    #377,DONE       ;SET DONE
9522  044710  000207                            RTS     PC
```

```
                        ;;*****************************************************************
                        ;SBTTL  TIME OUT PROCESSOR ROUTINE
                        ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
                        ;*GATHERING DUTIES.
                        ;;*****************************************************************
044712                  TOPROC:
044712  104407                  SAVREG
044714  013702  003026          MOV     RKBAS,R2
044720  012701  001174          MOV     #$REGS,R1       ;SET UP R1 FOR RK REGISTER STORAGE
044724  016221  000000          MOV     RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
044730  016221  000010          MOV     RKCS2(R2),(R1)+ ;REGISTERS
044734  016221  000020          MOV     RKDC(R2),(R1)+
044740  016221  000006          MOV     RKDA(R2),(R1)+
044744  016221  000002          MOV     RKWC(R2),(R1)+
044750  016221  000004          MOV     RKBA(R2),(R1)+
044754  016221  000016          MOV     RKASOF(R2),(R1)+
044760  016221  000012          MOV     RKDS(R2),(R1)+
044764  016221  000014          MOV     RKER(R2),(R1)+
044770  005000                  CLR     R0              ;THIS CODE WILL ATTEMPT TO
                                                        ;RETRIEVE THE STATUS FROM THE
                                                        ;DRIVE.
044772  012705  002704          MOV     #PARM1,R5       ;SET UP TO USE PARM1
044776  112765  000141  000001  MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
045004  004737  040774          JSR     PC,DRVCAL       ;CALL DRIVER
045010  032765  000100  000014  BIT     #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
045016  001403                  BEQ     1$              ;NO - SKIP
045020  052737  002000  005474  BIS     #TWOTOS,RECODE  ;SET TWO TIMEOUTS FLAG
045026  062705  000040          1$:     ADD     #P.ADD,R5       ;BUMP R5 TO POINT TO DRIVE STATUS
045032  012521                  MOV     (R5)+,(R1)+     ;MOVE ALL THE DRIVE STATUS INTO THE
045034  012521                  MOV     (R5)+,(R1)+     ;TEMP REGS FOR REPORTING.
045036  012521                  MOV     (R5)+,(R1)+
045040  012521                  MOV     (R5)+,(R1)+
045042  012521                  MOV     (R5)+,(R1)+
045044  012521                  MOV     (R5)+,(R1)+
045046  012521                  MOV     (R5)+,(R1)+
045050  012521                  MOV     (R5)+,(R1)+

045052  012705  002620          MOV     #PARM0,R5       ;RESTORE PARM 0
045056  104410                  RESREG
045060  000207                  RTS     PC


                        ;;*****************************************************************
                        ;SBTTL  READ HEADER 0 ROUTINE
                        ;;*****************************************************************
045062                  RDHD0:
045062  104407                  SAVREG
045064  016501  000026          MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
045070  016500  000052          MOV     P.B10(R5),R0    ;GET THE CYLINDER ADRS
045074  042700  160017          BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
045100  006200                  ASR     R0              ;OFF UNUSED BITS AND POSITION
045102  006200                  ASR     R0              ;FOR USE AS THE DESIRED
045104  006200                  ASR     R0              ;CYLINDER IN THE READ
```

```
 9579  045106  006200                          ASR     R0              ;HEADER COMMAND.
 9580  045110  012705  002704                  MOV     #PARM1,R5       ;SET JP TO USE P.B. 1
 9581  045114  010165  000004                  MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
 9582  045120  010065  000002                  MOV     R0,P.CYLN(R5)   ;SET CYL NO.
 9583  045124  132737  000020  003115          BITB    #B.CFMT,FORMAT  ;TEST PRESENT FMT AND CHANGE IT
 9584  045132  001404                          BEQ     1$              ;TO THE OPPOSITE. THIS WILL CAUSE
 9585  045134  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5) ;A READ OF SECT 0 HDR ON
 9586  045142  000403                          BR      2$              ;THE READ HDR COMMAND
 9587  045144  152765  000020  000007  1$:     BISB    #B.CFMT,P.CS1H(R5)
 9588  045152  112765  000125  000001  2$:     MOVB    #RDHEAD,P.CMND(R5) ;SET READ HDR COMMAND
 9589  045160  012737  000000  177776          MOV     #PRO,@#PSW      ;ALLOW INTERRUPTS
 9590  045166  004737  040774                  JSR     PC,DRVCAL       ;DO READ HDR
 9591  045172  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
 9592  045200  153765  003115  000007          BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
 9593  045206  012705  002620                  MOV     #PARM0,R5       ;RESTORE P.B. 0 ADDRESS
 9594  045212  104410                          RESREG                  ;RESTORE R0-R5
 9595  045214  000207                          RTS     PC              ;RETURN
 9596
 9597
 9598
 9599                                  ;;*****************************************************************
 9600                                  .SBTTL  GET PACK ADDRESS ROUTINE
 9601                                  ;;*****************************************************************
 9602  045216  016537  000030  001174  GTPKAD: MOV     P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
 9603  045224  005037  001176                  CLR     $REG6           ;CLEAR REGISTERS FOR
 9604  045230  005037  001200                  CLR     $REG7           ;TRACK & SECTOR STORAGE
 9605  045234  116537  000026  001200          MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
 9606  045242  116537  000027  001176          MOVB    P.DTS+1(R5),$REG6
 9607  045250  005737  001200                  TST     $REG7                   ;ADJUST THE ADDRESS CONTAINED IN
 9608                                                                          ;THE RK REGISTERS FOR THE AUTOMATIC
 9609  045254  001403                          BEQ     1$                      ;INCREMENT
 9610  045256  005337  001200                  DEC     $REG7
 9611  045262  000426                          BR      5$
 9612  045264  032765  010000  000016  1$:     BIT     #CFMT,P.CS1(R5)
 9613  045272  001404                          BEQ     2$
 9614  045274  012737  000023  001200          MOV     #19.,$REG7
 9615  045302  000403                          BR      3$
 9616  045304  012737  000025  001200  2$:     MOV     #21.,$REG7
 9617  045312  005737  001176          3$:     TST     $REG6
 9618  045316  001403                          BEQ     4$
 9619  045320  005337  001176                  DEC     $REG6
 9620  045324  000405                          BR      5$
 9621  045326  012737  000002  001176  4$:     MOV     #2,$REG6
 9622  045334  005337  001174                  DEC     $REG5
 9623  045340  000207                  5$:     RTS     PC
 9624
 9625
 9626
 9627                                  ;;*****************************************************************
 9628                                  .SBTTL  BUILD EXPECTED HEADER
 9629                                  ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
 9630                                  ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
 9631                                  ;*7 FOR REPORTING.
 9632                                  ;;*****************************************************************
 9633  045342  104407                  BLDEXH: SAVREG
 9634  045344  016537  000030  001174          MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
```

```
 9635  045352  016501  000026                    MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
 9636  045356  042701  174377                    BIC     #174377,R1       ;CLEAR ALL BUT TRACK BITS
 9637  045362  006201                            ASR     R1               ;AND SECTOR. SHIFT THE TRACK
 9638  045364  006201                            ASR     R1               ;OVER TO CONFORM TO HEADER FORMAT
 9639  045366  006201                            ASR     R1               ;CHECK THE FORMAT BIT AND
 9640                                                                     ;IF SET, SET THE HEADER FORMAT
 9641  045370  016537  000026  001176            MOV     P.DTS(R5),$REG6  ;BIT.
 9642  045376  042737  177740  001176            BIC     #177740,$REG6    ;CLEAR ALL BUT SECTOR
 9643  045404  060137  001176                    ADD     R1,$REG6         ;ADD TRACK AND SECTOR TOGETHER
 9644  045410  052737  140000  001176            BIS     #140000,$REG6    ;INSERT BSE BITS
 9645  045416  032765  010000  000016            BIT     #CFMT,P.CS1(R5)
 9646  045424  001403                            BEQ     23$
 9647  045426  052737  001000  001176            BIS     #1000,$REG6
 9648  045434  013737  001174  001200  23$:      MOV     $REG5,$REG7      ;COMPUTE THE HEADER CRC
 9649  045442  013701  001176                    MOV     $REG6,R1
 9650  045446  043737  001176  001200            BIC     $REG6,$REG7
 9651  045454  043701  001174                    BIC     $REG5,R1
 9652  045460  050137  001200                    BIS     R1,$REG7
 9653  045464  104410                            RESREG
 9654  045466  000207                            RTS     PC
 9655
```

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

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

                        .SBTTL   *WATCH-DOG TIMER

                        ;;*****************************************************************
                        ;*
                        ;*      THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06 UNIBUS
                        ;*      SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
                        ;*      REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
                        ;*      THE RK06 DRIVER WILL USE THE LOCATION W.MTIM FOR
                        ;*      MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
                        ;*      WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
                        ;*      DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
                        ;*      (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
                        ;*      IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
                        ;*      TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
                        ;*      REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
                        ;*
                        ;*      THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
                        ;*      OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
                        ;*      THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
                        ;*      LIMIT FOR ALL OTHER COMMANDS.
                        ;*
                        ;*      FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
                        ;*      WATCH UP TO 8 OPERATIONS SIMULATEOUSLY. FOR SEQUENTIAL
                        ;*      OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
                        ;*
                        ;*CALL   JSR     PC,W.WTCH
                        ;*      RETURN          IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
                        ;*
                        ;*      OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
                        ;*      BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
                        ;*      IN THE PROGRAM DEVICE STATUS REGISTER OF THE
                        ;*      APPROPRIATE PARAMETER BLOCK WILL BE SET.
                        ;*
                        ;;*****************************************************************
                        
045470  010546          W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
045472  010446                  MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
045474  010346                  MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
045476  010246                  MOV     R2,-(SP)        ;SAVE R2 ON STACK
045500  013746  177776          MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
045504  005337  003046          DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
045510  001034                  BNE     20$             ;IF NOT ZERO RETURN
045512  013737  003050  003046  MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
045520  105737  003070          TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
045524  001426                  BEQ     20$             ;NO, RETURN
045526  013737  003032  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
045534  013702  003026          MOV     RKBAS,R2        ;LOAD BASE OF RK06 REGISTERS
045540  005337  003104          DEC     W.DRV           ;DECREMENT COMMAND TIMER
045544  001016                  BNE     20$             ;RETURN IF NO TIME OUT
```

```
                                        CLRB    W.TIME          ;RESET TIMING INDICATOR
                                        MOV     #BLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
                                                                ;  TABLE FOR INDEXING
                                        BIS     #CMDTO.P.PRST(R5) ;SET COMMAND TIME OUT
                                        CMP     R5,C.WAIT       ;CHECK IF DRIVER IS WAITING FOR
                                                                ;  COMMAND COMPLETION
                                        BNE     5$              ;NO, DO NOT ALTER WAITING FOR
                                                                ;  COMMAND COMPLETION
                                        CLR     C.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
                                5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
                                20$:    MOV     (SP)+,PS        ;RESTORE PSW
                                        MOV     (SP)+,R2        ;RESTORE R2
                                        MOV     (SP)+,R3        ;RESTORE R3
                                        MOV     (SP)+,R4        ;RESTORE R4
                                        MOV     (SP)+,R5        ;RESTORE R5
                                        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
                        ;*
                        ;;*****************************************************************

045620  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
045622  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
045624  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
045626  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
045630  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
045632  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
045634  013702  003026          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
045640  016237  000010  002772  MOV     RKCS2(R2),T.CS2 ;STORE CS2
045646  032737  001000  002772  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
045654  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
045656  052737  100000  003042  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
045664  004737  051076          JSR     PC,R.CONT       ;REPORT ERROR
```

```
9784  045670  000137  050050                  JMP     I.RTRN          ;RETURN
9785
9786  045674  105737  003064          1$:     TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
9787  045700  001410                          BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
9788  045702  100403                          BMI     5$              ;CHECK IF RELEASE COMMAND
9789  045704  105037  003064                  CLRB    I.ISRL          ;YES,CLEAR FLAG
9790  045710  000473                          BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
9791
9792  045712  105037  003064          5$:     CLRB    I.ISRL          ;CLEAR FLAG
9793  045716  000137  047032                  JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
9794
9795  045722  032737  010400  002772  6$:     BIT     #NED!JFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
9796                                                                  ;  UNIT FIELD ERROR
9797  045730  001413                          BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
9798  045732  013704  002772                  MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
9799  045736  042704  177770                  BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
9800  045742  013705  003102                  MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
9801  045746  016237  000000  002770          MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
9802  045754  000137  046242                  JMP     I.ERRC          ;REPORT ERROR
9803
9804  045760  016237  000012  003010  7$:     MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
9805  045766  032737  000001  003010          BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
9806                                                                  ;  PORT
9807  045774  001041                          BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
9808
9809                                          ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
9810  045776  032737  164000  002772          BIT     #DLT!WCE!UPE!NEM,T.CS2
9811
9812  046004  001007                          BNE     10$             ;INDICATE ERROR
9813  046006  016237  000014  003006          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
9814
9815                                  ;       CHECK FOR DATA TRANSFER ERROR TYPE ERROR
9816  046014  032737  125700  003006          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
9817
9818  046022  001407                          BEQ     11$             ;NO, WAIT FOR RELEASE OF RKCS DRIVE
9819
9820  046024  052737  000010  003042  10$:    BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
9821  046032  004737  051076                  JSR     PC,R.CONT       ;REPORT ERROR
9822  046036  000137  050050                  JMP     I.RTRN          ;RESTORE REGISTERS
9823
9824  046042  105037  003070          11$:    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
9825  046046  005037  003104                  CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
9826  046052  013705  003102                  MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
9827                                                                  ;  ADDRESS
9828  046056  052765  010000  000014          BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
9829                                                                  ;  PROGRAM DRIVE STATUS REGISTER
9830  046064  005037  003044                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
9831  046070  004737  051052                  JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
9832  046074  000137  050050                  JMP     I.RTRN          ;GO RESTORE REGISTERS
9833
9834  046100  013705  003044          I.IOO:  MOV     O.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
9835  046104  001002                          BNE     2$              ;IS COMMAND WAITING PROCESSING
9836                                                                  ;  YES, DO PROCESSING
9837  046106  000137  047032                  JMP     I.ATTN          ;NO, PROCESS ATTENTION
9838
9839  046112  013704  002772          2$:     MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
9840  046116  042704  177770            BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
9841
9842
9843  046122  126504  000000            CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
9844  046126  001401                     BEQ     3$              ;YES, CONTINUE
9845  046130  000000                     HALT                    ;NO, DRIVER ERROR
9846  046132  122765  000164  000001 3$: CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
9847  046140  001002                     BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
9848  046142  000137  046500            JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
9849
9850  046146  005037  003044       10$: CLR     @.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
9851  046152  005037  003104            CLR     @.DRV           ;CLEAR WATCH-DOG TIME
9852  046156  105037  003070            CLRB    @.TIME          ;RESET TIMING ON THIS DRIVE
9853  046162  016237  000000  002770      MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
9854  046170  032737  100000  002770      BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
9855  046176  001021                     BNE     I.ERRC          ;YES, PROCESS ERROR
9856  046200  016237  000016  003004      MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9857  046206  133737  003071  003005      BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
9858  046214  001004                     BNE     15$             ;YES, REPORT ERROR
9859  046216  004737  051064            JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
9860  046222  000137  050050            JMP     I.RTRN          ;RESTORE REGISTERS
9861
9862  046226  052765  000010  000014 15$: BIS    #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
9863
9864  046234  004737  050520       I.ERRA: JSR  PC,I.CSTS       ;STORE CONTROLLER STATUS
9865  046240  000405                     BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
9866
9867  046242  013765  002770  000016 I.ERRC: MOV  T.CS1,P.CS1(R5) ;GET ERROR RKCS1
9868  046250  004737  050542            JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
9869  046254  016265  000032  000062 I.ERR: MOV   RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
9870  046262  016265  000030  000060      MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
9871  046270  004037  050066            JSR     R0,I.CCLR       ;CLEAR CONTROLLER
9872  046274  050050                     I.RTRN                  ;ERROR RETURN
9873  046276  032765  010400  000020      BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
9874                                                            ;  UNIT FIELD ERROR
9875  046304  001046                     BNE     5$              ;YES, REPORT ERROR
9876  046306  004037  050624            JSR     R0,I.STAT       ;GATHER DRIVE STATUS
9877  046312  050050                     I.RTRN                  ;ERROR RETURN
9878  046314  112737  000005  002770      MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
9879  046322  004037  050150            JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
9880  046326  050050                     I.RTRN                  ;ERROR RETURN
9881  046330  133737  003071  003005      BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
9882  046336  001407                     BEQ     2$              ;NO, INDICATE DRIVE ERROR
9883  046340  052737  000020  003042      BIS     #E.CLFT,E.CONT  ;SET ATTENTION DID NOT RESET
9884                                                            ;  WITH CLEAR
9885  046346  004737  051076            JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
9886  046352  000137  050050            JMP     I.RTRN          ;GO RESTORE REGISTERS
9887
9888  046356  032737  040000  003014 2$: BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
9889  046364  001403                     BEQ     3$              ;YES, CHECK FAULT
9890  046366  052765  000040  000014      BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
9891  046374  032737  001000  003016 3$: BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
9892  046402  001407                     BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
9893  046404  052737  002000  003042      BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
9894  046412  004737  051076            JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
9895  046416  000137  050050            JMP     I.RTRN          ;RETURN
```

```

046422  032765  000020  003014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
046430  001017                          BNE     10$                ;YES, GO REPORT ERROR
046432  032737  020000  003014          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS CYCLING DOWN
046440  001413                          BEQ     10$                ;NO, REPORT ERROR
046442  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
046450  113737  003071  003070          MOVB    INTMSK,W.TIME      ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
046456  013737  003054  003104          MOV     W.8SEC,W.DRV
046464  000137  050050                  JMP     I.RTRN             ;GO RESTORE REGISTERS

046470  004737  051052          10$:    JSR     PC,R.ABNL          ;GO REPORT ERROR
046474  000137  050050                  JMP     I.RTRN             ;GO RESTORE REGISTERS

                                .SBTTL  *READ ALL HEADERS INTERRUPT SEQUENCE

046500  016237  000000  002770  I.HDAL: MOV     RKCS1(R2),T.CS1    ;STORE CS1 TO CHECK CONTROLLER
                                                                   ;  ERROR
046506  032737  100000  002770          BIT     #CERR,T.CS1        ;CHECK IF CONTROLLER ERROR
046514  001422                          BEQ     5$                 ;NO, CHECK FOR ATTENTION

046516  005037  003044                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETE
046522  105037  003070                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
046526  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
046532  013765  002770  000016          MOV     T.CS1,P.CS1(R5)    ;STORE ERROR RKCS1
046540  004737  050742                  JSR     PC,I.CST1          ;STORE CONTROLLER REGISTERS
046544  004037  050066                  JSR     R0,I.CCLR          ;CLEAR CONTROLLER
046550  050050                          I.RTRN                     ;ERROR RETURN
046552  004737  051052                  JSR     PC,R.ABNL          ;INDICATE ERROR RETURN
046556  000137  050050                  JMP     I.RTRN             ;RESTORE REGISTERS

046562  016537  000016  003004  5$:     MOV     RKASOF(R5),T.ASOF  ;STORE ATTENTION SUMMARY
046570  133737  003071  003005          BITB    INTMSK,T.ASOF+1    ;CHECK IF DRIVE ATTENTION IS SET
046576  001410                          BEQ     7$                 ;NO, CHECK IF READ ALL HEADERS
046600  005037  003044                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETION
046604  105037  003070                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
046610  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
046614  000137  046234                  JMP     I.ERRA             ;GO REPORT ERROR

046620  013701  003060          7$:     MOV     HDR.AD,R1          ;GET MAIN MEMORY ADDRESS
046624  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET FIRST WORD OF HEADER
046630  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET SECOND WORD OF HEADER
046634  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET THIRD WORD OF HEADER
046640  010137  003060                  MOV     R1,HDR.AD          ;STORE ADDRESS FOR NEXT HEADER
046644  016237  000010  002772          MOV     RKCS2(R2),T.CS2    ;STORE CS2 TO CHECK FOR DATA LATE
046652  032737  100000  002772          BIT     #DLT,T.CS2         ;CHECK FOR DATA LATE
046660  001055                          BNE     35$                ;YES, REPORT ERROR
046662  005337  003062                  DEC     HDR.CT             ;DECREMENT NUMBER OF HEADER YET TO READ
046666  001026                          BNE     25$                ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
046670  005037  003044                  CLR     O.WAIT             ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
046674  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT FOR THIS DRIVE
046700  105037  003070                  CLRB    W.TIME             ;CLEAR WATCH DOG TIME ON THIS DRIVE
046704  012762  000003  000026          MOV     #3,RKMR1(R2)       ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
046712  112737  000001  002770          MOVB    #DR.SEL,T.CS1      ;LOAD SELECT COMMAND
046720  004037  050150                  JSR     R0,I.ISSU          ;GET SECTOR COUNT
046724  050050                          I.RTRN                     ;ERROR RETURN
046726  013765  003016  000056          MOV     T.MR3,P.B11(R5)    ;LOAD SECTOR COUNT
```

```
 9952  046734  004737  051064                    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
 9953  046740  000137  050050                    JMP     I.RTRN          ;RESTORE REGISTERS
 9954
 9955  046744  016562  000002  000020  25$:      MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
 9956  046752  016562  000004  000006            MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
 9957  046760  116565  000007  000017            MOVB    F.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 9958  046766  042765  165777  000016            BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
 9959                                                    ;  DRIVE TYPE
 9960  046774  112765  000125  000016            MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
 9961  047002  016562  000016  000000            MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
 9962  047010  000137  050050                    JMP     I.RTRN          ;RESTORE REGISTERS
 9963
 9964  047014  052737  000400  003042  35$:      BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
 9965  047022  004737  051076                    JSR     PC,R.CONT       ;REPORT ERROR
 9966  047026  000137  050050                    JMP     I.RTRN          ;RESTORE REGISTERS
 9967
 9968                                  .SBTTL   *DRIVE ATTENTION SCANNER
 9969
 9970  047032  016237  000000  002770  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
 9971                                                    ;  REGISTER 1 FOR COMPARISON
 9972  047040  032737  100000  002770            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
 9973  047046  001441                            BEQ     5$              ;NO, CHECK IF ATTENTION
 9974
 9975                                          ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
 9976  047050  032737  164000  002772            BIT     #DLT!WCE!UPE!NEM,T.CS2
 9977
 9978  047056  001007                            BNE     1$              ;INDICATE ERROR
 9979  047060  016237  000014  003006            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
 9980
 9981                                  ;         CHECK FOR DATA TRANSFER ERROR TYPE
 9982  047066  032737  125700  003006            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
 9983
 9984  047074  001407                            BEQ     2$              ;NO DATA TRANSFER ERROR
 9985
 9986  047076  052737  000010  003042  1$:       BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
 9987  047104  004737  051076                    JSR     PC,R.CONT       ;REPORT ERROR
 9988  047110  000137  050050                    JMP     I.RTRN          ;RESTORE REGISTERS
 9989
 9990  047114  013704  002772            2$:     MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
 9991  047120  042704  177770                    BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
 9992  047124  105037  003070                    CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
 9993  047130  005037  003104                    CLR     W.DRV           ;RESET TIMER VALUE
 9994  047134  013705  003102                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
 9995
 9996                                  ;         CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
 9997                                  ;           IN PROGRAM DEVICE STATUS REGISTER
 9998  047140  042765  000006  000014            BIC     #DRVPOS!DRVPDT,P.PRST(R5)
 9999
10000  047146  000137  046242                    JMP     I.ERRC          ;GO REPORT ERROR
10001
10002  047152  032737  040000  002770  5$:       BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
10003  047160  001002                            BNE     6$              ;YES, PROCESS INTERRUPT
10004  047162  000137  050050                    JMP     I.RTRN          ;RESTORE REGISTERS
10005
10006  047166  016237  003016  003004  6$:       MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10007  047174  105737  003005                    TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
10008  047200  001007                        BNE     7$              ;YES GO PROCESS INTERRUPT
10009  047202  052737  000002  003042        BIS     #E.NOAT,E.CONT  ;SET NO ATTENTION IN ATTENTION SUMMARY
10010  047210  004737  051076                JSR     PC,R.CONT       ;GO REPORT ERROR
10011  047214  000137  050050                JMP     I.RTRN          ;GO RESTORE REGISTERS
10012
10013  047220  133737  003071  003005  7$:   BITB    INTMSK,T.ASOF+1 ;CHECK IF DESIRED INTERRUPT
10014  047226  001007                        BNE     8$              ;YES, GO PROCESS IT
10015  047230  052737  000004  003042        BIS     #E.UATT,E.CONT  ;SET UNSOLICATED ATTENTION
10016  047236  004737  051076                JSR     PC,R.CONT       ;GO REPORT ERROR
10017  047242  000137  050050                JMP     I.RTRN          ;GO RESTORE REGISTERS
10018
10019  047246  013705  003102          8$:   MOV     PBLKT,R5        ;STORE PARAMETER BLOCK TABLE
10020  047252  116504  000000                MOVB    P.DRVN(R5),R4   ;STORE DRIVE NUMBER
10021  047256  032765  020000  000014        BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
10022  047264  001402                        BEQ     11$             ;NO, CONTINUE
10023  047266  000137  047750                JMP     I.UNLD          ;SERVICE DRIVE IN POSITION AFTER ERROR
10024
10025  047272  042765  000002  000014  11$:   BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
10026  047300  005062  000026                CLR     RKMR1(R2)       ;CLEAR MAINTENANCE REGISTER 1
10027  047304  112737  000001  002770        MOVB    #DR.SEL,T.CS1   ;LOAD COMMAND
10028  047312  004037  050150                JSR     R0,I.ISSU       ;SELECT DRIVE WITH ATTENTION HIGH
10029  047316  050050                        I.RTRN                  ;ERROR RETURN
10030  047320  013765  003016  000042        MOV     T.MR3,P.BOO(R5) ;STORE STATUS BYTE 00 MESS B
10031  047326  032765  000200  000042        BIT     #S.FLT,P.BOO(R5) ;CHECK IF DRIVE FAULT
10032  047334  001401                        BEQ     12$             ;NO, CHECK FOR DRIVE STATUS CHANGE
10033  047336  000461                        BR      I.AERR          ;PROCESS ERROR
10034
10035  047340  013765  003014  000040  12$:   MOV     T.MR2,P.ADD(R5) ;STORE MAINTENANCE REGISTER 2
10036  047346  032765  040000  000040        BIT     #S.DSC,P.ADD(R5) ;CHECK FOR DRIVE STATUS CHANGE
10037  047354  001004                        BNE     13$             ;YES, PROCESS DRIVE STATUS CHANGE
10038  047356  052765  004000  000014        BIS     #NODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
10039  047364  000446                        BR      I.AERR          ;PROCESS ERROR
10040
10041  047366  112737  000005  002770  13$:   MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
10042  047374  004037  050150                JSR     R0,I.ISSU       ;CLEAR DRIVE STATUS CHANGE
10043  047400  050050                        I.RTRN                  ;ERROR RETUN
10044  047402  013765  003004  000032        MOV     T.ASOF,P.ASOF(R5) ;STORE ATTENTION SUMMARY
10045  047410  133765  003071  000033        BITB    INTMSK,P.ASOF+1(R5) ;CHECK IF ATTENTION RESET
10046  047416  001407                        BEQ     15$             ;YES, CONTINUE INTERRUPT PROCESSING
10047  047420  052737  000020  003042        BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
10048                                                                ;  WITH DRIVE CLEAR
10049  047426  004737  051076                JSR     PC,R.CONT       ;FLAG ERROR
10050  047432  000137  050050                JMP     I.RTRN          ;RESTORE REGISTERS
10051
10052  047436  013765  003014  000040  15$:   MOV     T.MR2,P.ADD(R5) ;STORE MAINTENANCE REGISTER 2
10053  047444  032765  040000  000040        BIT     #S.DSC,P.ADD(R5) ;CHECK IF DRIVE STATUS CHANGE
10054                                                                ;  RESET
10055  047452  001404                        BEQ     16$             ;YES, CONTINUE INTERRUPT PROCESSING
10056  047454  052765  000040  000014        BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
10057  047462  000407                        BR      I.AERR          ;GO PROCESS ERROR
10058
10059  047464  105037  003070          16$:   CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
10060  047470  005037  003104                CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
10061  047474  004737  051064                JSR     PC,R.NORM       ;REPORT SUCCESSFUL COMMAND COMPLETION
10062  047500  000563                        BR      I.RTRN          ;RESTORE REGISTERS
10063
```

```
                                .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.BSEC,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
                                BIS     #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
```

```
 10120  050040  005037  003104                CLR     W.DRV           ;CLEAR TIME COUNT
 10121  050044  004737  051052                JSR     PC,R.ABNL       ;REPORT ERROR
 10122
 10123  050050  012600                I.RTRN: MOV     (SP)+,R0        ;RESTORE R0
 10124  050052  012601                        MOV     (SP)+,R1        ;RESTORE R1
 10125  050054  012602                        MOV     (SP)+,R2        ;RESTORE R2
 10126  050056  012603                        MOV     (SP)+,R3        ;RESTORE R3
 10127  050060  012604                        MOV     (SP)+,R4        ;RESTORE R4
 10128  050062  012605                        MOV     (SP)+,R5        ;RESTORE R5
 10129  050064  000002                        RTI                     ;RETURN
 10130
```

```
10131                                   .SBTTL   *CONTROLLER CLEAR ROUTINE
10132
10133                                   ;;*****************************************************************
10134                                   ;*
10135                                   ;*      THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
10136                                   ;*      AND CHECK IF THE CONTROLLER ERRORS ARE RESET.  IF THE ERROR IS NOT
10137                                   ;*      CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
10138                                   ;*      E.CCLR SET IN E.CONT.
10139                                   ;*
10140                                   ;*      REGISTER        USE
10141                                   ;*      --------        ---
10142                                   ;*
10143                                   ;*      R2              ADDRESS OF RK06 REGISTERS
10144                                   ;*      R5              ADDRESS OF PARAMETER BLOCK
10145                                   ;*
10146                                   ;*CALL  JSR     R0,I.CCLR
10147                                   ;*      <ADDRESS OF ERROR RETURN>
10148                                   ;*      RETURN
10149                                   ;*
10150                                   ;*****************************************************************
10151
10152  050066  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
10153  050074  016237  000000  002770          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
10154  050102  032737  100000  002770          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
10155                                                                  ;  CLEAR ERROR
10156  050110  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
10157  050112  052737  000001  003042          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
10158  050120  004737  051076                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
10159  050124  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
10160  050126  000200                          RTS     R0              ;RETURN
10161
10162  050130  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10163  050136  112737  177777  003064          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
10164  050144  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
10165  050146  000200                          RTS     R0              ;RETURN
```

```
10166                                   .SBTTL   *COMMAND ISSUED BY DRIVER SERVICE ROUTINE
10167
10168                                   ;;*****************************************************************
10169                                   ;*
10170                                   ;*      THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
10171                                   ;*      AND CHECK IF A CONTROLLER ERROR OCCURRED.  IF A CONTROLLER
10172                                   ;*      ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
10173                                   ;*      CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
10174                                   ;*      ADDRESS IN A.CONT.
10175                                   ;*
10176                                   ;*      REGISTER        USE
10177                                   ;*      --------        ---
10178                                   ;*
10179                                   ;*      R2              ADDDRESS OF RK06 REGISTERS
10180                                   ;*      R5              ADDRESS OF PARAMETER BLOCK
10181                                   ;*
10182                                   ;*CALL  JSR     R0,I.ISSU
10183                                   ;*      <ADDRESS OF ERROR RETURN>
10184                                   ;*      RETURN
10185                                   ;*
10186                                   ;*      ROUTINES USED:
10187                                   ;*      --------------
10188                                   ;*
10189                                   ;*              I.CCLR
10190                                   ;*              I.STOR
10191                                   ;*
10192                                   ;;*****************************************************************
10193
10194  050150  013746  002770          I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
10195  050154  005037  002772                  CLR     T.CS2           ;CLEAR TEMPORARY CS2
10196  050160  116537  000000  002772          MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
10197  050166  013762  002772  000010          MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
10198  050174  116537  000007  002771          MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
10199  050202  142737  177753  002771          BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
10200                                                                  ;  FORMAT AND DRIVE TYPE
10201  050210  013762  002770  000000          MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
10202  050216  105762  000000          1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
10203  050222  100375                          BPL     1$
10204  050224  004737  050372                  JSR     PC,I.STOR       ;GO STORE REGISTERS
10205  050230  032737  100000  002770          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
10206  050236  001437                          BEQ     5$              ;NO, RETURN
10207  050240  032737  001000  002772          BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
10208  050246  001406                          BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
10209  050250  052737  100000  003042          BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
10210  050256  004737  051076                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
10211  050262  000440                          BR      10$             ;RETURN
10212
10213                                                   ;CHECK IF ANY CONTROLLER ERROR IS SET
10214  050264  032737  024000  002770  2$:     BIT     #CTO!SPAR,T.CS1
10215  050272  001027                          BNE     7$
10216  050274  032737  176400  002772          BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
10217  050302  001023                          BNE     7$
10218  050304  032737  131761  003006          BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
10219  050312  001017                          BNE     7$
10220
10221  050314  122716  000005                  CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
10222  050320  001003                        BNE     3$                      ;NO, DO NOT SET DRIVE HARD ERROR
10223  050322  052765  000020  000014        BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
10224  050330  004037  050066          3$:   JSR     R0,I.CCLR               ;GO ISSUE A CONTROLLER CLEAR
10225  050334  050364                        I0$                             ;ERROR RETURN
10226  050336  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)           ;SET INTERRUPT ENABLE
10227  050344  005726                        TST     (SP)+                   ;ADJUST STACK
10228  050346  005720                        TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
10229  050350  000200                        RTS     R0                      ;RETURN
10230
10231  050352  052737  001000  003042  7$:   BIS     #E.CERR,E.CONT  ;SET CONTROLLER ERROR DURING
10232                                                                ;  DRIVER SERVICING
10233  050360  004737  051076                JSR     PC,R.CONT               ;REPORT ERROR
10234  050364  005726                  10$:  TST     (SP)+                   ;ADJUST STACK
10235  050366  011000                        MOV     (R0),R0                 ;ADJUST R0 FOR ERROR RETURN
10236  050370  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
                        ;*
                        ;;**********************************************************************

016237 000000 002770  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
016237 000010 002772          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
016237 000002 002774          MOV     RKWC(R2),T.WCR
016237 000004 002776          MOV     RKBA(R2),T.BA
016237 000006 003000          MOV     RKDA(R2),T.DA
016237 000012 003010          MOV     RKDS(R2),T.DS
016237 000014 003006          MOV     RKER(R2),T.ER
016237 000016 003004          MOV     RKASOF(R2),T.ASOF
016237 000020 003002          MOV     RKDCYL(R2),T.DC
016237 000026 003012          MOV     RKMR1(R2),T.MR1
016237 000034 003014          MOV     RKMR2(R2),T.MR2
016237 000036 003016          MOV     RKMR3(R2),T.MR3
016237 000030 003020          MOV     RKECPS(R2),T.POS
016237 000032 003022          MOV     RKECPT(R2),T.PAT
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
                        ;*
                        ;*****************************************************************

 050520  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
                                                                         ;OF LAST COMMAND ISSUED
 050526  042737  000036  002770          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
 050534  053765  002770  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
 050542  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
 050550  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
 050556  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
 050564  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
 050572  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
 050600  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
 050606  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
                                                                         ;  OFFSET
 050614  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
 050622  000207                          RTS     PC                      ;RETURN
```

```
                        *GATHER DRIVE STATUS

                        .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
                        ;*
                        ;;*****************************************************************

 050624  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                         ;  FOR STATUS BYTE 01
 050632  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 050640  004037  050150                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
 050644  051034                          3$                              ;ERROR RETURN
 050646  013765  003014  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
 050654  013765  003016  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
 050662  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                         ;  FOR STATUS BYTE 10
 050670  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 050676  004037  050150                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
 050702  051034                          3$                              ;ERROR RETURN
 050704  013765  003014  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
 050712  013765  003016  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
 050720  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
                                                                         ;  FOR STATUS BYTE 11
 050726  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 050734  004037  050150                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
 050740  051034                          3$                              ;ERROR RETURN
 050742  013765  003014  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
 050750  013765  003016  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
 050756  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
                                                                         ;  FOR STATUS BYTE 00
 050762  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
 050770  004037  050150                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
 050774  051034                          3$                              ;ERROR RETURN
 050776  013765  003014  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
 051004  013765  003016  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
 051012  032737  001000  003016          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
 051020  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
103770  051022  052737  002000  000342          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
103776  061030  004737  051076                  JSR     PC,R.CONT               ;REPORT ERROR
104002  061034  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
104004  051036  000200                          RTS     R0                      ;RETURN
104006
104006  051040  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
104014  061046  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
104016  051050  000200                          RTS     R0                      ;RETURN
104020
```

F16


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

051120  010546          C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
051122  010446                  MOV     R4,-(SP)        ;STORE R4 ON STACK
051124  010346                  MOV     R3,-(SP)        ;STORE R3 ON STACK
051126  010246                  MOV     R2,-(SP)        ;STORE R2 ON STACK
051130  010146                  MOV     R1,-(SP)        ;STORE R1 ON STACK
051132  010046                  MOV     R0,-(SP)        ;STORE R0 ON STACK
051134  013746  177776          MOV     PS,-(SP)        ;STORE PSW ON STACK
051140  013737  003032  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
051146  017605  000016          MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
051152  062766  000002  000016  ADD     #2,16(SP)       ;ADJUST RETURN
051160  016504  000000          MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
051164  042704  177770          BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
051170  010537  003102          MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
051174  116437  003072  003071  MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
051202  116437  003072  003070  MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
051210  013737  003052  003104  MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

051216  013702  003026          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
```

```
10450  051222  042765  075176  000014          BIC     #!C<DRVUSE!W.WCK!NOCHK!DRPDRV!D"BAII>,P.PRST(R5)
10451
10452  051230  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
10453  051232  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
10454  051236  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
10455  051240  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
10456
10457  051244  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
10458  051246  020001                          CMP     R0,R1           ;CHECK IF FINISHED
10459  051250  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
10460  051252  105037  003064                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
10461  051256  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
10462  051262  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
10463  051266  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
10464  051274  001402                          BEQ     3$              ;NO, PROCESS
10465  051276  000137  052012                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
10466
10467  051302  122765  000107  000000.  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
10468                                                                  ;  START SPINDLE
10469                                                                  ;  RECALIBRATE
10470                                                                  ;  OFFSET
10471                                                                  ;  SEEK
10472                                                                  ;  UNLOAD
10473
10474  051310  101174                          BHI     25$             ;NO, DRIVE COMMAND
10475                                                                  ;  SELECT DRIVE
10476                                                                  ;  PACK ACKNOWLEDGE
10477                                                                  ;  CLEAR
10478
10479  051312  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
10480  051320  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
10481                                                                  ;  READ DATA
10482                                                                  ;  WRITE DATA
10483                                                                  ;  READ HEADER
10484                                                                  ;  WRITE HEADER
10485                                                                  ;  WRITE CHECK
10486  051322  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10487  051330  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
10488  051336  005037  003044                  CLR     @.WAIT          ;CLEAR WAIT FOR COMMAND
10489  051342  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
10490  051350  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
10491  051352  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10492  051360  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10493  051366  000431                          BR      8$              ;GO ISSUE COMMAND
10494
10495  051370  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
10496  051376  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
10497  051400  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
10498  051406  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
10499  051414  000416                          BR      8$              ;GO ISSUE COMMAND
10500
10501  051416  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
10502  051424  001003                          BNE     7$              ;NO, CHECK IF RECAL
10503  051426  013737  003056  003104          MOV     W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
10504  051434  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
10505  051442  001003                          BNE     8$              ;NO, CONTINUE
```

```
105506  051444  013737  003054  003104          MOV     W.BSEC,W.DRV     ;LOAD RECAL TIME FOR 8 SECONDS
105507  051452  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
105508  051460  042765  165777  003016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
105509                                                  ;  AND DRIVE TYPE
105510  051466  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
105511  051474  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
105512  051502  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
105513  051510  001533                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
105514  051512  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
105515  051520  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
105516  051526  004737  045470          10$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
105517  051532  016237  000000  002770          MOV     RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REGISTER 1
105518  051540  032737  000200  002770          BIT     #RDY,T.CS1       ;WAIT FOR READY
105519  051546  001767                          BEQ     10$
105520  051550  032737  100000  002770          BIT     #CERR,T.CS1      ;CHECK FOR ERROR
105521  051556  001011                          BNE     15$              ;YES, GIVE NORMAL RETURN
105522  051560  004737  045470          11$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
105523  051564  016237  000016  003004          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
105524  051572  133737  003071  003005          BITB    INTMSK,T.ASOF+1  ;CHECK IF INTERRUPT HAS OCCURRED
105525  051600  001767                          BEQ     11$              ;WAIT FOR DRIVE INTERRUPT
105526  051602  105037  003070          15$:    CLRB    W.TIME           ;RESET TIMING ON THIS DRIVE
105527  051606  005037  003104                  CLR     W.DRV            ;CLEAR DRIVE TIMING COUNT
105528  051612  004737  051064                  JSR     PC,R.NORM        ;INDICATE COMMAND IS FINISHED
105529  051616  000137  052772                  JMP     C.RTRN           ;RESTORE REGISTERS
105530
105531  051622  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
105532  051630  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
105533  051636  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
105534  051644  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
105535  051652  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
105536  051660  001010                          BNE     25$              ;NO, GO ISSUE THE COMMAND
105537  051662  032765  000200  000014          BIT     #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
105538  051670  001404                          BEQ     25$              ;NO, GO ISSUE THE COMMAND
105539  051672  012765  000123  000016          MOV     #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
105540  051700  000406                          BR      26$              ;GO ISSUE COMMAND
105541
105542  051702  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
105543  051710  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
105544  051716  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
105545  051724  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!A.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
105546                                                  ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
105547                                                  ;  BITS 16-17
105548  051732  010537  003044                  MOV     R5,Q.WAIT        ;LOAD WAITING FOR COMMAND
105549  051736  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
105550  051744  001403                          BEQ     27$              ;NO, LOAD CS2
105551  051746  052765  000020  000020          BIS     #BAI,P.CS2(R5)   ;SET INHIBIT BUS ADDRESS INCREMENT
105552  051754  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
105553  051762  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
105554  051770  001403                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
105555  051772  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
105556  052000  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
105557  052006  000137  052772                  JMP     C.RTRN           ;RESTORE REGISTERS
105558
105559                                  .SBTTL  *SPECIAL COMMAND PROCESSING
105560
105561  052012  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
105562  052020  001132                      BNE     10$             ;NO, PROCESS OTHER COMMANDS
105563  016562  000020  000010              MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
105564  116565  000007  000017              MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
105565  042765  165777  000016              BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
                                                                        ;  AND DRIVE TYPE
105566  052044  112765  000001  000016      MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
105567  016562  000016  000000              MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
105570  004737  054470                  2$: JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
105571  016265  000000  000016              MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
105572  032765  000200  000016              BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
105573  001767                              BEQ     2$
105574  004737  050542                      JSR     PC,I.CSTI       ;STORE CONTROLLER REGISTERS
105575  016265  000034  000040              MOV     RKMR2(R2),P.AOD(R5) ;STORE STATUS BYTE OO MESSAGE A
105576  016265  000036  000042              MOV     RKMR3(R2),P.BOC(R5) ;STORE STATUS BYTE OO MESSAGE B
105577  032765  100000  000016              BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
105600  001436                              BEQ     6$              ;NO, GATHER DRIVE STATUS
105601  105037  003070                      CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
105602  005037  003104                      CLR     W.DRV           ;CLEAR WATCH DOG COUNT
105603  032765  000400  000014              BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
105604  001043                              BNE     8$              ;YES, INDICATE NORMAL RETURN
105605  032765  001000  000020              BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
105606  001043                              BNE     9$              ;YES, INDICATE CONTROLLER ERROR
105607  004037  050066                      JSR     RO,I.CCLR       ;CLEAR ERROR
105610  052772                          ;.RTRN                      ;ERROR RETURN
105611  032765  010400  000020              BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
105612  001007                              BNE     5$              ;REPORT ERROR
105613  032765  000001  000036              BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
105614  001003                              BNE     5$              ;YES, REPORT ERROR
105615  052765  010000  000014              BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
105616  004737  051052                  5$: JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
105617  000137  052772                      JMP     C.RTRN          ;RESTORE REGISTERS
105620
105621  004037  050624                  6$: JSR     RO,I.STAT       ;GATHER DRIVE STATUS
105622  052772                          C.RTRN                      ;ERROR RETURN
105623  105037  003070                      CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
105624  005037  003104                      CLR     W.DRV           ;RESET WATCH-DOG TIME
105625  032765  000400  000014              BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
105626  001402                              BEQ     8$              ;NO, REPORT ERROR
105627  005062  000000                      CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
105630  004737  051064                  8$: JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
105631  000137  052772                      JMP     C.RTRN          ;RESTORE REGISTERS
105632
105633  052737  100000  003042  9$:     BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
105634  004737  051076                      JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
105635  000137  052772                      JMP     C.RTRN
105636
105637  022765  000140  000001  10$:     CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
105640  001040                              BNE     13$             ;NO, CHECK IF READ ALL HEADERS
105641  010537  003044                      MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
105642                                                              ;  WAIT FOR COMMAND
105643  052765  000010  000020              BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
105644  016562  000020  000010              MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
105645  112737  000001  003064              MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
105646  116565  000007  000017              MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
105647  042765  165777  000016              BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
105650                                                              ;  AND DRIVE TYPE
```

```
052360  112765  000101  000016         MOVB    #SEL.DRV,P.CS1(R5)  ;STORE COMMAND
052366  032765  000400  000014         BIT     #NOCHK,P.PRST(R5)  ;CHECK IF NO CHECK MODE
052374  001403                         BEQ     11$                ;NO, DO NOT RESET INTERRUPT ENABLE
052376  042765  000100  000016         BIC     #IE,P.CS1(R5)      ;RESET INTERRUPT ENABLE
052404  016562  000016  000000  11$:   MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
052412  000137  052772                 JMP     C.RTRN             ;RESTORE REGISTERS
052416  122765  000164  000001  13$:   CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
052424  001053                         BNE     30$                ;NO, CHECK IF CONTROLLER CLEAR
052426  010537  003044                 MOV     R5,O.WAIT          ;SET WAITING FOR COMMAND COMMPLETION
052432  016537  000010  003060         MOV     P.BALO(R5),HDR.AD  ;LOAD HEADER ADDRESS
052440  132765  000020  000007         BITB    #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
052446  001404                         BEQ     14$                ;YES, LOAD 22 IN HEADER COUNT
052450  012737  000024  003062         MOV     #20.,HDR.CT        ;LOAD 20 IN SECTOR COUNT
052456  000403                         BR      22$                ;GO ISSUE READ HEADER COMMAND
052460  012737  000026  003062  14$:   MOV     #22.,HDR.CT        ;LOAD 22 IN SECTOR COUNT
052466  016562  000002  000020  22$:   MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
052474  016562  000004  000006         MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
052502  016562  000020  000010         MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
052510  116565  000007  000017         MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
052516  042765  165777  000016         BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
                                                                   ;  AND FORMAT
052524  112765  000125  000016         MOVB    #RDHEAD,P.CS1(R5)  ;STORE READ HEADER COMMAND
052532  032765  000400  000014         BIT     #NOCHK,P.PRST(R5)  ;CHECK IF NO CHECK MODE
052540  001027                         BNE     34$                ;YES, INDICATE ILLEGAL DRIVER COMMAND
052542  016562  000016  000000         MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
052550  000137  052772                 JMP     C.RTRN             ;RESTORE REGISTERS
052554  122765  000176  000001  30$:   CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
052562  001012                         BNE     32$                ;NO, CHECK IF SUBSYSTEM CLEAR
052564  004037  050066                 JSR     R0,I.CCLR          ;CLEAR CONTROLLER
052570  052772                         C.RTRN                     ;ERROR RETURN
052572  032765  000400  000014         BIT     #NOCHK,P.PRST(R5)  ;CHECK IF NO CHECK MODE
052600  001472                         BEQ     40$                ;NO, INDICATE NORMAL RETURN
052602  005062  000000                 CLR     RKCS1(R2)          ;RESET INTERRUPT ENABLE
052606  000467                         BR      40$                ;INDICATE NORMAL RETURN
052610  122765  000177  000001  32$:   CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
052616  001406                         BEQ     36$                ;YES, CLEAR SUBSYSTEM
052620  052737  000100  003042  34$:   BIS     #E.ILLD,E.CONT     ;SET ILLEGAL DRIVER COMMAND
052626  004737  051076                 JSR     PC,R.CONT          ;REPORT ERROR
052632  000457                         BR      C.RTRN             ;RESTORE REGISTERS
052634  012762  000040  000010  36$:   MOV     #SCLR,RKCS2(R2)    ;ISSUE SUBSYSTEM CLEAR
052642  016265  000000  000016         MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
052650  032765  100000  000016         BIT     #CERR,P.CS1(R5)    ;CLEAR IF CONTROLLER ERROR RESET
052656  001406                         BEQ     37$                ;NO, FINISH COMMAND
052660  052737  000001  003042         BIS     #BIT0,E.CONT       ;SET CLEAR SUBSYSTEM DID NOT CLEAR
                                                                   ;  CONTROLLER ERROR
052666  004737  051076                 JSR     PC,R.CONT          ;REPORT ERROR
052672  000437                         BR      C.RTRN             ;RESTORE REGISTERS
052674  013746  003050          37$:   MOV     W.MILI,-(SP)       ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
                                                                   ;  TO DISAPPEAR
052700  016265  000000  000016  38$:   MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
10674  052706  032765  040000  000016          BIT     #DI.P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
10675  052714  001411                          BEQ     39$             ;YES, FINISH COMMAND
10676  052716  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
10677  052720  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
10678  052722  005726                          TST     (SP)+           ;ADJUST STACK
10679  052724  052737  000040  003042          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10680                                                                  ;  DRIVE ATTENTIONS
10681  052732  004737  051076                  JSR     PC,R.CONT       ;REPORT ERROR
10682  052736  000415                          BR      C.RTRN          ;RESTORE REGISTER
10683
10684  052740  005726                  39$:    TST     (SP)+           ;ADJUST STACK
10685  052742  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10686  052750  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
10687  052752  112737  177777  003064          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
10688  052760  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10689  052766  004737  051064          40$:    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
10690
10691  052772  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
10692  052776  012600                          MOV     (SP)+,R0        ;RESTORE R0
10693  053000  012601                          MOV     (SP)+,R1        ;RESTORE R1
10694  053002  012602                          MOV     (SP)+,R2        ;RESTORE R2
10695  053004  012603                          MOV     (SP)+,R3        ;RESTORE R3
10696  053006  012604                          MOV     (SP)+,R4        ;RESTORE R4
10697  053010  012605                          MOV     (SP)+,R5        ;RESTORE R5
10698  053012  000207                          RTS     PC              ;RETURN
10699                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
10700
10701                                  ;;**************************************************************
10702                                  ;*
10703                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10704                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
10705                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10706                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
10707                                  ;*
10708                                  ;*CALL
10709                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
10710                                  ;*      JSR     PC,OCTBIN
10711                                  ;*      <ADDRESS OF ERROR RETURN>
10712                                  ;*      RETURN
10713                                  ;*
10714                                  ;;**************************************************************
10715
10716  053014  010046          OCTBIN: MOV     R0,-(SP)        ;SAVE R0
10717  053016  010146                  MOV     R1,-(SP)        ;SAVE R1
10718  053020  010246                  MOV     R2,-(SP)        ;SAVE R2
10719  053022  016600  000010          MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
10720  053026  005001                  CLR     R1              ;CLEAR DATA WORDS
10721  053030  005002                  CLR     R2
10722  053032  112046          2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
10723  053034  001423                  BEQ     3$              ;IF ZERO GET OUT
10724  053036  121627  000054          CMPB    (SP),#',        ;CHECK IF COMMA
10725  053042  001420                  BEQ     3$              ;IF COMMA GET OUT
10726  053044  122716  000060          CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
10727  053050  003030                  BGT     4$              ;  AN OCTAL DIGIT
10728  053052  122716  000067          CMPB    #'7,(SP)
10729  053056  002425                  BLT     4$
```

```
10730  053060  006301                  ASL     R1              ; *2
10731  053062  006102                  ROL     R2
10732  053064  006301                  ASL     R1              ; *4
10733  053066  006102                  ROL     R2
10734  053070  006301                  ASL     R1              ; *8
10735  053072  006102                  ROL     R2
10736  053074  042716  177770          BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
10737  053100  062601                  ADD     (SP)+,R1        ;ADD THIS DIGIT
10738  053102  000753                  BR      2$              ;LOOP
10739  053104  005726          3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
10740  053106  010166  000010          MOV     R1,10(SP)       ;SAVE RESULT
10741  053112  010237  053146          MOV     R2,$HIOCT
10742  053116  012602                  MOV     (SP)+,R2        ;RESTORE R2
10743  053120  012601                  MOV     (SP)+,R1        ;RESTORE R1
10744  053122  012600                  MOV     (SP)+,R0        ;RESTORE R0
10745  053124  062716  000002          ADD     #2,(SP)         ;ADJUST RETURN
10746  053130  000207                  RTS     PC              ;RETURN
10747
10748  053132  005726          4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
10749  053134  012602                  MOV     (SP)+,R2        ;RESTORE R2
10750  053136  012601                  MOV     (SP)+,R1        ;RESTORE R1
10751  053140  012600                  MOV     (SP)+,R0        ;RESTORE R0
10752  053142  013616                  MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
10753  053144  000207                  RTS     PC              ;GO PROCESS ERROR
10754  053146  000000  $HIOCT: .WORD   0                       ;HIGH ORDER BITS GO HERE
10755                   .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
10756
10757                   ;;*****************************************************************
10758                   ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
10759                   ;*WITH A RANGE OF 0 TO 2(+33)-1.
10760                   ;*CALL:
10761                   ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
10762                   ;*      RETURN                  ;;RETURN HERE THE RANDOM
10763                   ;*                              ;;NUMBER WILL BE IN
10764                   ;*                              ;;$HINUM,$LONUM
10765
10766  053150          $RAND:
10767  053150  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10768  053152  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10769  053154  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10770  053156  013700  053250  MOV     $LONUM,R0       ;;SET R0 WITH LOW
10771  053162  013701  053246  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
10772  053166  012702  177771  MOV     #-7,R2          ;;SET SHIFT COUNT
10773  053172  006300          1$:     ASL     R0              ;;SHIFT R0 LEFT AND
10774  053174  006101                  ROL     R1              ;;ROTATE CARRY INTO R1 AND
10775  053176  005202                  INC     R2              ;;CHECK FOR DONE
10776  053200  001374                  BNE     1$              ;;CONTINUE SHIFT LOOP
10777  053202  063700  053250          ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
10778  053206  005501                  ADC     R1              ;;PROPOGATE CARRY
10779  053210  063701  053246          ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
10780  053214  062700  001057          ADD     #1057,R0        ;;ADD LOW CONSTANT
10781  053220  005501                  ADC     R1              ;;PROPOGATE CARRY
10782  053222  062701  047401          ADD     #47401,R1       ;;ADD HIGH CONSTANT
10783  053226  010037  053250          MOV     R0,$LONUM       ;;SAVE R0
10784  053232  010137  053246          MOV     R1,$HINUM       ;;SAVE R1
10785  053236  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                        RTS     PC              ;;RETURN
$HINUM: .WORD   176543
$LONUM: .WORD   123456
.SBTTL  TYPE ROUTINE

;;**********************************************************************
;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;*
;*CALL:
;*1) USING A TRAP INSTRUCTION
;*       TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;*OR
;*       TYPE
;*       MESADR
;*

$TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
        BPL     1$              ;;BR IF YES
        HALT                    ;;HALT HERE IF NO TERMINAL
        BR      3$              ;;LEAVE
1$:     MOV     R0,-(SP)        ;;SAVE R0
        MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
        BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
        BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
        MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
        JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
61$:    .WORD   0               ;;MESSAGE ADDRESS
62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
        BNE     60$             ;;YES,SKIP TYPE OUT
2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
60$:    MOV     (SP)+,R0        ;;RESTORE R0
3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
        RTI                     ;;RETURN
4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
        BEQ     8$
        CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
        BNE     5$
        TST     (SP)+           ;;POP  <CR><LF> EQUIV
        TYPE                    ;;TYPE A CR AND LF
        $CRLF
        CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
        BR      2$              ;;GET NEXT CHARACTER
5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
        BNE     2$              ;;IF NO GO GET NEXT CHAR.
        MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
                                                        ;;AND THE NULL CHAR.
[illegible]  105366  000001        7$:    DECB   1(SP)      ;;DOES A NULL NEED TO BE TYPED?
[illegible]  001775                BLT    6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
[illegible]  004737  053464        JSR    PC,$TYPEC         ;;GO TYPE A NULL
[illegible]  105337  053530        DECB   $CHARCNT          ;;DO NOT COUNT AS A COUNT
[illegible]  000770                BR     7$                ;;LOOP

                                   ;HORIZONTAL TAB PROCESSOR

[illegible]  112716  000040        8$:    MOVB   #' ,(SP)   ;;REPLACE TAB WITH SPACE
[illegible]  004737  053464        9$:    JSR    PC,$TYPEC  ;;TYPE A SPACE
[illegible]  132737  000007 053530        BITB   #7,$CHARCNT ;;BRANCH IF NOT AT
[illegible]  001002                        BNE    9$         ;;TAB STOP
[illegible]  005726                        TST    (SP)+      ;;POP SPACE OFF STACK
[illegible]  000764                        BR     2$         ;;GET NEXT CHARACTER
[illegible]  105777  125460        $TYPEC: TSTB   @$TPS      ;;WAIT UNTIL PRINTER IS READY
[illegible]  100375                        BPL    $TYPEC
[illegible]  116677  000002 125452         MOVB   2(SP),@$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
[illegible]  122766  000015 000002         CMPB   #CR,2(SP   ;;IS CHARACTER A CARRIAGE RETURN?
[illegible]  001003                        BNE    1$         ;;BRANCH IF NO
[illegible]  005037  053530                CLRB   $CHARCNT   ;;YES--CLEAR CHARACTER COUNT
[illegible]  000406                        BR     $TYPEX     ;;EXIT
[illegible]  122766  000012 000002  1$:    CMPB   #LF,2(SP)  ;;IS CHARACTER A LINE FEED?
[illegible]  001402                        BEQ    $TYPEX     ;;BRANCH IF YES
[illegible]  105210                        INCB   (PC)+      ;;COUNT THE CHARACTER
[illegible]  000000                $CHARCNT:.WORD 0          ;;CHARACTER COUNT STORAGE
[illegible]  000207                $TYPEX: RTS    PC
```

```
;;*****************************************************************
.SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
;*      USES ALL REGISTERS (R0-R5)
;*
;*      ENTER WITH   JSR   PC,M.DPIM
;*                   MULTIPLIER IN R2-R3
;*                   MULTIPLICAND IN R4-R5
;*                   PRODUCT RETURNED IN R0-R1-R2-R3
;*
;;*****************************************************************
M.DPIM: CLR     R0              ;CLEAR HI ORDER WORDS
        CLR     R1
        MOV     #41,-(SP)       ;MOVE 33 (DEC) TO COUNTER
M.DP01: ROR     R0
        ROR     R1
        ROR     R2              ;SHIFT TO ADD
        ROR     R3
        BCC     M.DP02          ;NO CARRY NO ADD
        ADD     R5,R1
        ADC     R0              ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
        ADD     R4,R0           ;  PRODUCT
M.DP02: DEC     @SP             ;DECREMENT COUNTER
        BNE     M.DP01
        TST     (SP)+           ;REMOVE THE COUNTER
```

```
053572 000207                          RTS     PC

                            ;;**************************************************************
                            .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
                            ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
                            ;*              USES ALL REGISTERS (R0-R5)
                            ;*
                            ;*              ENTER WITH  JSR   PC,M.DPID
                            ;*                      DIVIDEND IN R0-R1-R2-R3
                            ;*                      DIVISOR IN R4-R5
                            ;*                      REMAINDER RETURNED IN R0-R1
                            ;*                      QUOTIENT RETURNED IN R2-R3
                            ;*
                            ;;**************************************************************
053574 012746 000040        M.DPID: MOV     #40,-(SP)               ;COUNTER FOR DIVISION CYCLES
053600 010446                       MOV     R4,-(SP)                ;HI ORDER
053602 010546                       MOV     R5,-(SP)                ;LO ORDER DIVISOR TO THE STACK
053604 005466 000002                NEG     2(SP)                   ;FORM NEGATIVE
053610 005416                       NEG     @SP                     ;   VERSION OF THE DIVISOR
053612 005666 000002                SBC     2(SP)
053616 061601                       ADD     @SP,R1
053620 005500                       ADC     R0                      ;PERFORM THE INITIAL SUBTRACTION
053622 066600 000002                ADD     2(SP),R0
053626 103445                       BCS     M.DP50                  ;IF CARRY THEN OVERFLOW HAS OCCURRED
053630 005046                       CLR     -(SP)                   ;THIS IS A LONGER LASTING CARRY BIT
053632 006103               M.DP40: ROL     R3
053634 006102                       ROL     R2
053636 006101                       ROL     R1
053640 006100                       ROL     R0
053642 005716                       TST     @SP                     ;TEST "CARRY INDICATOR"
053644 001410                       BEQ     M.DP41                  ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
053646 005016                       CLR     @SP                     ;CLEAR UP FOR NEXT TIME
053650 066601 000002                ADD     2(SP),R1
053654 005500                       ADC     R0                      ;ADD -(DIVISOR)
053656 005516                       ADC     @SP     ; I             ;SET "CARRY"
053660 066600 000004                ADD     4(SP),R0 ;<
053664 000404                       BR      M.DP42
053666 060501               M.DP41: ADD     R5,R1
053670 005500                       ADC     R0                      ;ADD +(DIVISOR)
053672 005516                       ADC     @SP     ; I             ;SET "CARRY"
053674 060400                       ADD     R4,R0   ;<
053676 005516               M.DP42: ADC     @SP                     ;SET "CARRY"
053700 005716                       TST     @SP                     ;TEST THE UPDATE INDICATOR
053702 001401                       BEQ     .+4     ;>              ;IF ZERO FORGET IT
053704 005203                       INC     R3      ; I             ;NO CARRY POSSIBLE HERE
053706 005366 000006                DEC     6(SP)   ;<              ;DECREMENT COUNTER
053712 003347                       BGT     M.DP40                  ;BR IF MORE TO DO
053714 006003                       ROR     R3
053716 103404                       BCS     M.DP44
053720 060501                       ADD     R5,R1
053722 005500                       ADC     R0
053724 060400                       ADD     R4,R0
053726 000241                       CLC
053730 006103               M.DP44: ROL     R3
053732 062706 000010                ADD     #10,SP                  ;ADJUST STACK BY 4 WORDS
```

```
                        DOUBLE-PRECISION DIVIDE SUBROUTINE

053736  000242                  CLV
053740  000207                  RTS     PC
053742  062706  000006  M.DP50: ADD     #6,SP
053746  000262                  SEV
053750  000207                  RTS     PC


                        .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

                        ;;*****************************************************************
                        ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                        ;*UNSIGNED OCTAL ASCIZ NUMBER.
                        ;*CALL
                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                        ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
                        ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

053752  104407          $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
053754  016601  000002          MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
053760  012705  054071          MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
053764  012704  000014          MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
053770  012703  177770          MOV     #^C7,R3         ;;MASK
053774  012100                  MOV     (R1)+,R0        ;;LOWER WORD
053776  012101                  MOV     (R1)+,R1        ;;HIGH WORD
054000  005002                  CLR     R2              ;;TERMINATOR
054002  110245          1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
054004  010002                  MOV     R0,R2           ;;GET THIS DIGIT
054006  005304                  DEC     R4              ;;COUNT THIS CHARACTER
054010  003007                  BGT     3$              ;;BR IF NOT THE LAST DIGIT
054012  001405                  BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
054014  005205                  INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
054016  010566  000002          MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
054022  104410                  RESREG                  ;;RESTORE ALL REGISTERS
054024  000207                  RTS     PC              ;;RETURN TO USER
054026  006203          2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
054030  006001          3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
054032  006000                  ROR     R0              ;;      THE NEXT OCTAL DIGIT
054034  006001                  ROR     R1
054036  006000                  ROR     R0
054040  006001                  ROR     R1
054042  006000                  ROR     R0
054044  040302                  BIC     R3,R2           ;;MASK OUT ALL JUNK
054046  062702  000060          ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
054052  000753                  BR      1$              ;;GO PUT IT IN THE DATA TABLE
054054  000016          $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

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

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

054072  104407                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
054074  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
054100  012700  054252                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
054104  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
054110  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
054112  012202                          MOV     (R2)+,R2
054114  012737  000012  054170          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
054122  012704  054202                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
054126  012705  054204                  MOV     #$TNPWR+2,R5
054132  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
054134  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
054136  005602                          SBC     R2
054140  161502                          SUB     (R5),R2
054142  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
054144  005203                          INC     R3              ;;ADD 1 TO PARTIAL
054146  000772                          BR      2$              ;;LOOP
054150  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
054152  005502                          ADC     R2
054154  062402                          ADD     (R4)+,R2
054156  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
054160  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
054164  110320                          MOVB    R3,(R0)+        ;;SAVE IT
054166  005327                          DEC     (PC)+           ;;DONE?
054170  000000                  4$:     .WORD   0
054172  001357                          BNE     1$              ;;BR IF NO
054174  105020                          CLRB    (R0)+           ;;TERMINATOR
054176  104410                          RESREG                  ;;RESTORE REGISTERS
054200  000207                          RTS     PC              ;;RETURN
054202  145000                  $TNPWR: 145000                  ;;1.0E09
054204  035632                          35632
054206  160400                          160400                  ;;1.0E08
054210  002765                          2765
054212  113200                          113200                  ;;1.0E07
054214  000230                          230
054216  041100                          041100                  ;;1.0E06
054220  000017                          17
054222  103240                          103240                  ;;1.0E05
054224  000001                          1
054226  023420                          23420                   ;;1.0E04
054230  000000                          0
054232  001750                          1750                    ;;1.0E03
054234  000000                          0
054236  000144                          144                     ;;1.0E02
054240  000000                          0
054242  000012                          12                      ;;1.0E01
054244  000000                          0
054246  000001                          1                       ;;1.0E00
054250  000000                          0
054252  000014                  $DECVL: .BLKB   12.              ;;RESERVE STORAGE FOR ASCIZ STRING
                        .SBTTL  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

054266  010046                  $SJPRS: MOV     R0,-(SP)        ;;SAVE R0
054270  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
054274  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
054276  001403                          BEQ     2$              ;;BR IF YES
054300  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
054304  001773                          BEQ     1$              ;;BR IF YES
054306  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
054310  010037  054316                  MOV     R0,3$           ;;SAVE FOR TYPING
054314  104401                          TYPE                    ;;GO TYPE
054316  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
054320  012600                          MOV     (SP)+,R0        ;;RESTORE R0
054322  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
054324  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

054326  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
054332  116637  000001  054551          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
054340  112637  054553                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
054344  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
054350  000406                          BR      $TYPON
054352  112737  000001  054551  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
054360  112737  000006  054553          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
054366  112737  000005  054550  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
054374  010346                          MOV     R3,-(SP)        ;;SAVE R3
054376  010446                          MOV     R4,-(SP)        ;;SAVE R4
054400  010546                          MOV     R5,-(SP)        ;;SAVE R5
```

```
 054432  113704  054553          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 054436  005404                  NEG     R4
 054440  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 054444  110437  054552          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 054450  113704  054551          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 054454  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 054460  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
 054462  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 054464  000404                  BR      3$              ;;GO DO MSB
 054466  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
 054470  006105                  ROL     R5
 054472  006105                  ROL     R5
 054474  010503                  MOV     R5,R3
 054476  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 054500  105337  054552          DECB    $OMODE          ;;TYPE THIS DIGIT?
 054504  100016                  BPL     7$              ;;BR IF NO
 054506  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
 054512  001002                  BNE     4$              ;;TEST FOR 0
 054514  005704                  TST     R4              ;;SUPPRESS THIS 0?
 054516  001403                  BEQ     5$              ;;BF IF YES
 054520  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 054522  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 054526  052703  000040  5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 054532  110337  054546          MOVB    R3,8$           ;;SAVE FOR TYPING
 054536  104401  054546          TYPE    ,8$             ;;GO TYPE THIS DIGIT
 054542  105337  054550  7$:     DECB    $OCNT           ;;COUNT BY 1
 054546  003347                  BGT     2$              ;;BR IF MORE TO DO
 054550  002402                  BLT     6$              ;;BR IF DONE
 054552  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 054554  000744                  BR      2$              ;;GO DO THE LAST DIGIT
 054556  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
 054560  012604                  MOV     (SP)+,R4        ;;RESTORE R4
 054562  012603                  MOV     (SP)+,R3        ;;RESTORE R3
 054564  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 054572  012616                  MOV     (SP)+,(SP)
 054574  000002                  RTI                     ;;RETURN
 054576     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 054577     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 054600     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 054601     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 054602  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
                         .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE


                         ;;****************************************************************
                         ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                         ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                         ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                         ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                         ;*REPLACED WITH SPACES.
                         ;*CALL:
                         ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                         ;*      TYPDS                   ;;GO TO THE ROUTINE


 054604                  $TYPDS:
 054604  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 054606  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
054560  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
054562  010346                          MOV     R3,-(SP)        ;;PLSH R3 ON STACK
054564  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
054566  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
054572  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
054576  100004                          BPL     1$              ;;BR IF INPUT IS POS.
054600  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
054602  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
054610  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
054612  012703  054770                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
054616  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
054622  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
054624  016001  054760                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
054630  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
054632  002402                          BLT     4$              ;;BR IF DONE
054634  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
054636  000774                          BR      3$
054640  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
054642  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
054644  001002                          BNE     5$              ;;FALL THROUGH IF 0
054646  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
054650  100407                          BMI     7$              ;;BR IF YES
054652  106316                  5$:     ASLB    (SP)            ;;MSD?
054654  103003                          BCC     6$              ;;BR IF NO
054656  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
054664  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
054670  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
054674  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
054676  005720                          TST     (R0)+           ;;JUST INCREMENTING
054700  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
054704  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
054706  003002                          BGT     8$              ;;GO TO EXIT
054710  010502                          MOV     R5,R2           ;;GET THE LSD
054712  000764                          BR      6$              ;;GO CHANGE TO ASCII
054714  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
054716  100003                          BPL     9$              ;;BR IF NO
054720  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
054726  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
054730  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
054732  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
054734  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
054736  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
054740  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
054742  104401  054770                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
054746  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
054754  012616                          MOV     (SP)+,(SP)
054756  000002                          RTI                     ;;RETURN TO USER
054760  023420                  $DTBL:  10000.
054762  001750                          1000.
054764  000144                          100.
054766  000012                          10.
054770  000004                  $DBLK:  .BLKW   4
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

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

055000                                  $ERROR:
055000  105237  001103                 7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
055004  001775                                 BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
055006  013777  001102  124126                 MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
055014  032777  002000  124116                 BIT     #BIT10,@SWR     ;;BELL ON ERROR?
055022  001402                                 BEQ     1$              ;;NO - SKIP
055024  104401  001310                         TYPE    ,$BELL          ;;RING BELL
055030  005237  001112                 1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
055034  011637  001116                         MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
055040  162737  000002  001116                 SUB     #2,$ERRPC
055046  117737  124044  001114                 MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
055054  032777  020000  124056                 BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
055062  001004                                 BNE     20$             ;;SKIP TYPEOUTS
055064  004737  041300                         JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
055070  104401  001315                         TYPE    ,$CRLF
055074                                  20$:
055074  122737  000001  001340                 CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
055102  001007                                 BNE     2$              ;;NO,SKIP APT ERROR REPORT
055104  113737  001114  055116                 MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
055112  004737  055760                         JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
055116     000                  21$:    .BYTE   0
055117     000                          .BYTE   0
055120  000777                  22$:    BR      22$             ;;APT ERROR LOOP
055122  005777  124012           2$:    TST     @SWR            ;;HALT ON ERROR
055126  100001                         BPL     3$              ;;SKIP IF CONTINUE
055130  000000                         HALT                    ;;HALT ON ERROR!
055132  032777  001000  124000   3$:    BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
055140  001402                         BEQ     4$              ;;BR IF NO
055142  013716  001110                 MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
055146  005737  001306           4$:    TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
055152  001402                         BEQ     5$              ;;BR IF NONE
055154  013716  001306                 MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
055160                           5$:
055160  022737  023600  000042         CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
055166  001001                         BNE     6$              ;;BRANCH IF NO
055170  000000                         HALT                    ;;YES
055172                           6$:
055172  000002                         RTI                     ;;RETURN
                                        .SBTTL  TTY INPUT ROUTINE

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

                                        .DSABL  LSB
```

```
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
055174  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
055176  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
055204  105777  123734          1$:     TSTB    @$TKS           ;;WAIT FOR
055210  100375                          BPL     1$              ;;A CHARACTER
055212  117766  123730  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
055220  042766  177600  000004          BIC     #^C(177),4(SP)  ;;GET RID OF JUNK IF ANY
055226  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
055234  001013                          BNE     3$              ;;BRANCH IF NO
055236  105777  123702          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
055242  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
055244  117746  123676                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
055250  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
055254  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
055260  001366                          BNE     2$              ;;IF NOT DISCARD IT
055262  000750                          BR      1$              ;;YES, RESUME
055264  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
055272  002407                          BLT     4$              ;;BRANCH IF YES
055274  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
055302  003003                          BGT     4$              ;;BRANCH IF YES
055304  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
055312  000002                  4$:     RTI                     ;;GO BACK TO USER
055314  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
055321     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
055326  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
055334  020075     000
055337     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
055344  036440  000040


                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;POWER DOWN ROUTINE
055350  012737  055362  000024  $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
055356  000000                          HALT                    ;HANG UP
055360  000776                          BR      .-2


                                ;POWER UP ROUTINE
055362  005037  055434          $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
055366  005237  055434          4$:     INC     $PWRCT
055372  001375                          BNE     4$
055374  012737  055350  000024          MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
055402  012737  000340  000026          MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
055410  012737  000340  000036          MOV     #PR7,TRAPVEC+2
055416  012706  001100                  MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
055422  104401  055436                  TYPE    ,PWRMSG         ;TYPE "POWER FAILED"
055426  000005                          RESET                   ;CLEAR THE UNIBLS
055430  000177  123452                  JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
11346  055434  000000                  $PWRCT: .WORD   0
11347  055436  005015  047520  042527  PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
11348  055444  020122  040506  046111
11349  055452  042105  005015     000
11350          055460                          .EVEN
11351
11352
11353
11354                                  ;;*****************************************************************
11355                                  ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
11356                                  ;*CALLED BY "SCOPER"
11357                                  ;;*****************************************************************
11358  055460  105737  001103          SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
11359  055464  001406                          BEQ     6$              ;BR IF NOT
11360  055466  032777  001000  123444          BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
11361  055474  001402                          BEQ     6$              ;BR IF NOT
11362  055476  013716  001110                  MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
11363  055502  000002                  6$:     RTI                     ;RETURN
11364
11365
11366
11367                                  .SBTTL  SCOPE HANDLER ROUTINE
11368
11369                                  ;;*****************************************************************
11370                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11371                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11372                                  ;*AND LOAD THE ERROR FLAG ($ERF_G) INTO DISPLAY<15:08>
11373                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11374                                  ;*SW14=1         LOOP ON TEST
11375                                  ;*SW11=1         INHIBIT ITERATIONS
11376                                  ;*SW09=1         LOOP ON ERROR
11377                                  ;*CALL
11378                                  ;*      SCOPE           ;;SCOPE=IOT
11379
11380  055504                          $SCOPE:
11381  055504  005737  001304                  TST     $TIMES          ;CHECK CURRENT ITERATION NUMBER
11382  055510  001404                          BEQ     $XTSTR          ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
11383  055512  032777  040000  123420  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11384  055520  001101                          BNE     $OVER           ;;YES IF SW14=1
11385                                  ;*****START OF CODE FOR THE XOR TESTER*****
11386  055522  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11387                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11388  055524  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11389  055530  012737  055550  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11390  055536  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
11391  055542  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11392  055546  000450                          BR      $SVLAD          ;;GO TO THE NEXT TEST
11393  055550  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11394  055552  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11395  055556  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
11396  055560                          6$:;*****END OF CODE FOR THE XOR TESTER*****
11397  055560  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11398  055564  001421                          BEQ     3$              ;;BR IF NO
11399  055566  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
11400  055574  101015                          BHI     3$              ;;BR IF NO
11401  055576  032777  001000  123334          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
```

```
11402  055604  001404                          BEQ     4$                ;;BR IF NO
11403  055606  013737  001110  001106  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
11404  055614  000443                          BR      $OVER
11405  055616  105037  001103          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
11406  055622  005037  001304                  CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11407  055626  000412                          BR      1$                ;;ESCAPE TO THE NEXT TEST
11408  055630  032777  004000  123302  3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
11409  055636  001006                          BNE     1$                ;;BR IF YES
11410  055640  005237  001104                  INC     $ICNT             ;;INCREMENT ITERATION COUNT
11411  055644  023737  001304  001104          CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
11412  055652  002024                          BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
11413  055654  012737  000001  001104  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
11414  055662  013737  055740  001304          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
11415  055670  105237  001102          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
11416  055674  113737  001102  001324          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
11417  055702  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
11418  055706  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
11419  055712  005037  001306                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11420  055716  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11421  055724  013777  001102  123210  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
11422  055732  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
11423  055736  000002                          RTI                       ;;FIXES PS
11424  055740  003720                  $MXCNT: 2000.                     ;;MAX. NUMBER OF ITERATIONS
11425                                  .SBTTL  APT COMMUNICATIONS ROUTINE
11426
11427                                  ;;*****************************************************************
11428  055742  112737  000001  056206  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
11429  055750  112737  000001  056204  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
11430  055756  000403                          BR      $ATYC
11431  055760  112737  000001  056206  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
11432  055766                          $ATYC:
11433  055766  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
11434  055770  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
11435  055772  105737  056204                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
11436  055776  001450                          BEQ     5$                ;;IF NOT:  BR
11437  056000  122737  000001  001340          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
11438  056006  001031                          BNE     3$                ;;IF NOT:  BR
11439  056010  132737  000100  001341          BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
11440  056016  001425                          BEQ     3$                ;;IF NOT:  BR
11441  056020  017600  000004                  MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
11442  056024  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
11443  056032  005737  001320          1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
11444  056036  001375                          BNE     1$                ;;IF NOT:  WAIT
11445  056040  010037  001334                  MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
11446  056044  105720                  2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
11447  056046  001376                          BNE     2$
11448  056050  163700  001334                  SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
11449  056054  006200                          ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
11450  056056  010037  001336                  MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
11451  056062  012737  000004  001320          MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
11452  056070  000413                          BR      5$
11453  056072  017637  000004  056116  3$:     MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
11454  056100  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
11455  056106  013746  177776                  MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
11456  056112  004737  053252                  JSR     PC,$TYPE          ;;CALL TYPE MACRO
11457  056116  000000                  4$:     .WORD   0
```

```
11458  056120                          5$:
11459  056120  105737  056206          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
11460  056124  001416                          BEQ     12$             ;;IF NOT:  BR
11461  056126  005737  001340                  TST     $ENV            ;;RUNNING UNDER APT?
11462  056132  001413                          BEQ     12$             ;;IF NOT:  BR
11463  056134  005737  001320          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
11464  056140  001375                          BNE     11$             ;;IF NOT:  WAIT
11465  056142  017637  000004  001322          MOV     @4.SP),$FATAL   ;;GET ERROR #
11466  056150  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
11467  056156  005237  001320                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
11468  056162  105037  056206          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
11469  056166  105037  056205                  CLRB    $LFLG           ;;CLEAR LOG FLAG
11470  056172  105037  056204                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
11471  056176  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11472  056200  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11473  056202  000207                          RTS     PC              ;;RETURN
11474  056204     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
11475  056205     000          $LFLG:  .BYTE   0               ;;LOG FLAG
11476  056206     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
11477          056210                  .EVEN
11478          000200          APTSIZE=200
11479          000001          APTENV=001
11480          000100          APTSPOOL=100
11481          000040          APTCSUP=040
11482                          .SBTTL  ROUTINE TO SIZE MEMORY
11483
11484                          ;;**************************************************************
11485                          ;*CALL:
11486                          ;*      JSR     PC,$SIZE
11487                          ;*      RETURN
11488                          ;*$LSTAD WILL CONTAIN:
11489                          ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11490                          ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11491                          ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11492                          ;*$KT11 IS THE MEMORY MANAGEMENT KEY
11493                          ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11494                          ;*      MUST BE SETUP BEFORE THE CALL
11495                          ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11496                          ;*      DETERMINED BY ROUTINE
11497
11498  056210  010046          $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
11499  056212  010146                  MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11500  056214  010246                  MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
11501  056216  010346                  MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
11502  056220  013746  0000C4          MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
11503  056224  013746  000006          MOV     @#ERRVEC+2,-(SP)
11504  056230  010600                  MOV     SP,R0           ;;SAVE THE STACK POINTER
11505                          ;;SET THE ERRVEC PS TO THE PRESENT PS
11506  056232  104400                  TRAP                    ;;PUSH OLD PSW AND PC ON STACK
11507  056234  012637  000006          MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
11508  056240  012701  003776          MOV     #3776,R1        ;;SETUP ADDRESS
11509  056244  105727                  TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
11510  056246  000200          $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
11511  056250  100062                  BPL     $CORE           ;;BR IF NO
11512  056252  012737  056410  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
11513  056260  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
```

```
11514  056264  052737  100000  056246          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
11515  056272  005046                          CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
11516  056274  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
11517  056300  012703  000010                  MOV     #10,R3          ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
11518  056304  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
11519  056312  011622                          MOV     (SP),(R2)+      ;;LOAD "PAR"
11520  056314  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
11521  056320  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
11522  056322  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
11523  056326  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
11524  056330  012737  056346  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
11525  056336  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
11526  056344  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
11527  056346  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
11528  056350  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
11529  056354  012737  056400  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
11530  056362  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
11531  056366  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
11532  056372  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
11533  056376  101371                          BHI     4$              ;;NO--TRY IT
11534  056400  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
11535  056402  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
11536  056406  000421                          BR      $SIZEX
11537  056410  042737  100000  056246  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
11538  056416  012737  056446  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
11539  056424  005002                          CLR     R2              ;;SET UP BANK
11540  056426  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
11541  056432  062702  000040                  ADD     #40,R2          ;;1K STEP
11542  056436  005711                          TST     (R1)            ;;TRAP ON TIME OUT
11543  056440  022701  177776                  CMP     #177776,R1      ;;LAST ONE
11544  056444  001370                          BNE     1$              ;;NO--TRY AGAIN
11545  056446  162701  004000          $CROUT: SUB     #4000,R1
11546  056452  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
11547  056456  010006                          MOV     R0,SP           ;;RESTORE THE STACK
11548  056460  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
11549  056464  012637  000004                  MOV     (SP)+,@#ERRVEC
11550  056470  010137  056512                  MOV     R1,$LSTAD       ;;LAST ADDRESS
11551  056474  010237  056514                  MOV     R2,$LSTBK       ;;LAST BANK
11552  056500  012603                          MOV     (SP)+,R3        ;;RESTORE R3
11553  056502  012602                          MOV     (SP)+,R2        ;;RESTORE R2
11554  056504  012601                          MOV     (SP)+,R1        ;;RESTORE R1
11555  056506  012600                          MOV     (SP)+,R0        ;;RESTORE R0
11556  056510  000207                          RTS     PC
11557  056512  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
11558  056514  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
11559                          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11560
11561                          ;;******************************************************************
11562                          ;*SAVE R0-R5
11563                          ;*CALL:
11564                          ;*      SAVREG
11565                          ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11566                          ;*
11567                          ;*TOP---(+16)
11568                          ;* +2---(+18)
11569                          ;* +4---R5
```

```
11570                                   ;* +6---R4
11571                                   ;* +8---R3
11572                                   ;*+10---R2
11573                                   ;*+12---R1
11574                                   ;*+14---R0
11575
11576  056516                           $SAVREG:
11577  056516  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11578  056520  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11579  056522  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11580  056524  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11581  056526  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11582  056530  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11583  056532  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11584  056536  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11585  056542  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11586  056546  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11587  056552  000002                          RTI
11588
11589                                   ;*RESTORE R0-R5
11590                                   ;*CALL:
11591                                   ;*      RESREG
11592  056554                           $RESREG:
11593  056554  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11594  056560  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11595  056564  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11596  056570  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11597  056574  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11598  056576  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11599  056600  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11600  056602  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11601  056604  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11602  056606  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11603  056610  000002                          RTI
11604                                   .SBTTL  TRAP DECODER
11605
11606                                   ;;**************************************************************
11607                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11608                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11609                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11610                                   ;*GO TO THAT ROUTINE.
11611
11612  056612  010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
11613  056614  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
11614  056620  005740                          TST     -(R0)           ;;BACKUP BY 2
11615  056622  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
11616  056624  006300                          ASL     R0              ;;POSITION FOR INDEXING
11617  056626  016000  056646                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
11618  056632  000200                          RTS     R0              ;;GO TO ROUTINE
11619
11620
11621                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
11622
11623  056634  011646                   $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
11624  056636  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
11625  056644  000002                          RTI                     ;;RESTORE THE PSW
```

```
                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
056646  056634                  $TRPAD: .WORD   $TRAP2
056650  053252                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
056652  054352                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
056654  054326                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
056656  054366                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
056660  054554                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

056662  055174                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
056664  056516                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
056666  056554                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
056670  055460                          SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE

056672  023040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
056700  052123  000040
056704  025052  020040     000  AS2SP2: .ASCIZ  /**  /
056711     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PARITY ERROR/
056716  020123  040520  044522
056724  054524  042440  051122
056732  051117     000
056735     116  047117  042455  EM2:    .ASCIZ  /NON-EXISTANT MEMORY/
056742  044530  052123  047101
056750  020124  042515  047515
056756  054522     000
056761     116  047117  042455  EM3:    .ASCIZ  /NON-EXISTANT DRIVE/
056766  044530  052123  047101
056774  020124  051104  053111
057002  000105
057004  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERROR/
057012  042511  042114  042440
057020  051122  051117     000
057025     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
057032  020123  044524  042515
057040  052517  000124
057044  020104  047524  041440  EM6:    .ASCIZ  /D TO C PARITY ERROR/
057052  050040  051101  052111
057060  020131  051105  047522
057066  000122
057070  051104  053111  020105  EM7:    .ASCIZ  /DRIVE DETECTED PARITY ERROR/
057076  042504  042524  052103
057104  042105  050040  051101
057112  052111  020131  051105
057120  047522  000122
057124  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
057132     000
057133     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
057140  046040  051517  000123
057146  046111  042514  040507  EM12:   .ASCIZ  /ILLEGAL FUNCTION/
```

```
 11682  057154  020114  052506  041516
 11683  057162  044524  047117     000
 11684  057167     120  047522  051107  EM13:   .ASCIZ  /PROGRAMMING ERROR/
 11685  057174  046501  044515  043516
 11686  057202  042440  051122  051117
 11687  057210     000
 11688  057211     116  047117  042455  EM14:   .ASCIZ  /NON-EXISTANT FUNCTION/
 11689  057216  044530  052123  047101
 11690  057224  020124  052506  041516
 11691  057232  044524  047117     000
 11692  057237     104  044522  042526  EM15:   .ASCIZ  /DRIVE TYPE ERROR/
 11693  057244  052040  050131  020105
 11694  057252  051105  047522  000122
 11695  057260  047506  046522  052101  EM16:   .ASCIZ  /FORMAT ERROR/
 11696  057266  042440  051122  051117
 11697  057274     000
 11698  057275     127  044522  042524  EM17:   .ASCIZ  /WRITE LOCK ERROR/
 11699  057302  046040  041517  020113
 11700  057310  051105  047522  000122
 11701  057316  051104  053111  020105  EM20:   .ASCIZ  /DRIVE UNSAFE/
 11702  057324  047125  040523  042506
 11703  057332     000
 11704  057333     123  042505  020113  EM21:   .ASCIZ  /SEEK INCOMPLETE/
 11705  057340  047111  047503  050115
 11706  057346  042514  042524     000
 11707  057353     103  046131  047111  EM22:   .ASCIZ  /CYLINDER OVERFLOW/
 11708  057360  042504  020122  053117
 11709  057366  051105  046106  053517
 11710  057374     000
 11711  057375     111  046114  043505  EM23:   .ASCIZ  /ILLEGAL CYLINDER ADDRESS/
 11712  057402  046101  041440  046131
 11713  057410  047111  042504  020122
 11714  057416  042101  051104  051505
 11715  057424  000123
 11716  057426  051104  053111  020105  EM24:   .ASCIZ  /DRIVE OFF TRACK/
 11717  057434  043117  020106  051124
 11718  057442  041501  000113
 11719  057446  051104  053111  020105  EM25:   .ASCIZ  /DRIVE TIMING ERROR/
 11720  057454  044524  044515  043516
 11721  057462  042440  051122  051117
 11722  057470     000
 11723  057471     104  052101  020101  EM26:   .ASCIZ  /DATA LATE/
 11724  057476  040514  042524     000
 11725  057503     103  047117  051124  EM27:   .ASCIZ  /CONTROLLER TIMEOUT/
 11726  057510  046117  042514  020122
 11727  057516  044524  042515  052517
 11728  057524  000124
 11729  057526  050117  051105  052101  EM30:   .ASCIZ  /OPERATION INCOMPLETE/
 11730  057534  047511  020116  047111
 11731  057542  047503  050115  042514
 11732  057550  042524     000
 11733  057553     110  040505  042504  EM31:   .ASCIZ  /HEADER VRC ERROR/
 11734  057560  020122  051126  020103
 11735  057566  051105  047522  000122
 11736  057574  040504  040524  041440  EM32:   .ASCIZ  /DATA CHECK ERROR/
 11737  057602  042510  045503  042440
```

```
 11738  057610  051122  051117     000
 11739  057615     127  044522  042524  EM33:   .ASCIZ  /WRITE CHECK ERROR/
 11740  057622  041440  042510  045503
 11741  057630  042440  051122  051117
 11742  057636     000
 11743  057637     104  052101  020101  EM34:   .ASCIZ  /DATA MISCOMPARE/
 11744  057644  044515  041523  046517
 11745  057652  040520  042522     000
 11746  057657     116  020117  051104  EM35:   .ASCIZ  /NO DRIVE RESPONSE-UFE & NXD/
 11747  057664  053111  020105  042522
 11748  057672  050123  047117  042523
 11749  057700  052455  042506  023040
 11750  057706  047040  042130     000
 11751  057713     104  044522  042526  EM36:   .ASCIZ  /DRIVE ERROR WILL NOT CLEAR/
 11752  057720  042440  051122  051117
 11753  057726  053440  046111  020114
 11754  057734  047516  020124  046103
 11755  057742  040505  000122
 11756  057746  051104  053111  020105  EM37:   .ASCIZ  /DRIVE STATUS CHANGE WILL NOT CLEAR/
 11757  057754  052123  052101  051525
 11758  057762  041440  040510  043516
 11759  057770  020105  044527  046114
 11760  057776  047040  052117  041440
 11761  060004  042514  051101     000
 11762  060011     101  052124  047047  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
 11763  060016  041040  052125  047040
 11764  060024  020117  052123  052101
 11765  060032  051525  041440  040510
 11766  060040  042516  020105  051117
 11767  060046  043040  052501  052114
 11768  060054     000
 11769  060055     101  052124  047047  EM41:   .ASCIZ  /ATT'N BUT DRIVE NOT AVAILABLE/
 11770  060062  041040  052125  042040
 11771  060070  044522  042526  047040
 11772  060076  052117  040440  040526
 11773  060104  046111  041101  042514
 11774  060112     000
 11775  060113     101  052124  047047  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
 11776  060120  053440  042510  020116
 11777  060126  047516  020124  054105
 11778  060134  042520  052103  042105
 11779  060142     000
 11780  060143     105  051122  051117  EM43:   .ASCIZ  /ERROR GATHERING DRIVE STATUS/
 11781  060150  043440  052101  042510
 11782  060156  044522  043516  042040
 11783  060164  044522  042526  051440
 11784  060172  040524  052524  000123
 11785  060200  052515  052114  050101  EM52:   .ASCIZ  /MULTIPLE DRIVE SELECT/
 11786  060206  042514  042040  044522
 11787  060214  042526  051440  046105
 11788  060222  041505  000124
 11789  060226  042510  042101  051105  EM53:   .ASCIZ  /HEADER COMPARE ERROR/
 11790  060234  041440  046517  040520
 11791  060242  042522  042440  051122
 11792  060250  051117     000
 11793  060253     123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
```

```
11794  060260  020123  044524  042515
11795  060266  052517  000124
11796  060272  051105  047522  020122  EM60:   .ASCIZ  /ERROR IN RECAL FOR RECOVERY/
11797  060300  047111  051040  041505
11798  060306  046101  043040  051117
11799  060314  051040  041505  053117
11800  060322  051105  000131
11801  060326  051120  043517  040522  EM61:   .ASCIZ  /PROGRAM ABORTING FATAL ERROR IN RETRY/
11802  060334  020115  041101  051117
11803  060342  044524  043516  043040
11804  060350  052101  046101  042440
11805  060356  051122  051117  044440
11806  060364  020116  042522  051124
11807  060372  000131
11808  060374  054503  044514  042116  EM62:   .ASCIZ  /CYLINDER MISCOMPARE/
11809  060402  051105  046440  051511
11810  060410  047503  050115  051101
11811  060416  000105
11812  060420  046103  040505  020122  EM63:   .ASCIZ  /CLEAR CONTROLLER DID NOT CLEAR ERROR/
11813  060426  047503  052116  047522
11814  060434  046114  051105  042040
11815  060442  042111  047040  052117
11816  060450  041440  042514  051101
11817  060456  042440  051122  051117
11818  060464     000
11819  060465     116  020117  052101  EM64:   .ASCIZ  /NO ATT'N IN ATT'N SUMMARY REGISTER/
11820  060472  023524  020116  047111
11821  060500  040440  052124  047047
11822  060506  051440  046525  040515
11823  060514  054522  051040  043505
11824  060522  051511  042524  000122
11825  060530  047125  047523  044514  EM65:   .ASCIZ  /UNSOLICITED ATTENTION/
11826  060536  044503  042524  020104
11827  060544  052101  042524  052116
11828  060552  047511  000116
11829  060556  047125  054105  042520  EM66:   .ASCIZ  /UNEXPECTED DATA TYPE ERROR
11830  060564  052103  042105  042040
11831  060572  052101  020101  054524
11832  060600  042520  042440  051122
11833  060606  051117     000
11834  060611     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
11835  060616  042040  042111  047040
11836  060624  052117  051040  051505
11837  060632  052105  053440  052111
11838  060640  020110  046103  040505
11839  060646  000122
11840  060650  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
11841  060656  041440  042514  051101
11842  060664  042040  042111  047040
11843  060672  052117  041440  042514
11844  060700  051101  042040  044522
11845  060706  042526  040440  052124
11846  060714  047047     000
11847  060717     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/
11848  060724  040514  042524  053440
11849  060732  042510  020116  047125
```

```
11850  060740  047514  042101  047111
11851  060746  020107  042510  042101
11852  060754  051105     000
11853  060757     103  047117  051124  EM72:   .ASCIZ  /CONTROLLER ERROR WHEN DRIVER SERVICING/
11854  060764  046117  042514  020122
11855  060772  051105  047522  020122
11856  061000  044127  047105  042040
11857  061006  044522  042526  020122
11858  061014  042523  053122  041511
11859  061022  047111  000107
11860  061026  051104  053111  020105  EM73:   .ASCIZ  /DRIVE DETECTED PARITY ERROR/
11861  061034  042504  042524  052103
11862  061042  042105  050040  051101
11863  061050  052111  020131  051105
11864  061056  047522  000122
11865  061062  047125  042504  044506  EM74:   .ASCIZ  /UNDEFINED ERROR/
11866  061070  042516  020104  051105
11867  061076  047522  000122
11868  061102  040515  045522  047111  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
11869  061110  020107  044124  051511
11870  061116  051440  041505  047524
11971  061124  020122  040502  000104
11872  061132  040502  020104  040504  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
11873  061140  040524  053040  051105
11874  061146  043111  047047  053440
11875  061154  052111  020110  042522
11876  061162  042101  020056  041505
11277  061170  020103  043117  046040
11878  061176  051501  020124  042522
11879  061204  051124  020131  051511
11880  061212  000072
11881  061214  042522  051124  020131  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
11882  061222  052523  041503  051505
11883  061230  043123  046125     000
11884  061235     122  052105  054522  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
11885  061242  052440  051516  041525
11886  061250  042503  051523  052506
11887  061256  000114
11888  061260  040503  047116  052117  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
11889  061266  043040  047111  020104
11890  061274  020101  040526  044514
11891  061302  020104  042510  042101
11892  061310  051105  044440  020116
11893  061316  051124  041501  020113
11894  061324  052512  052123  051040
11895  061332  040505  000104
11896  061336  040502  020104  042523  EM102:  .ASCIZ  /BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
11897  061344  052103  051117  042440
11898  061352  051122  051117  047440
11899  061360  020116  042523  052103
11900  061366  051117  047040  052117
11901  061374  046040  051511  042524
11902  061402  020104  040502  000104
11903  061410  044524  042515  026504  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
11904  061416  052517  020124  047117
11905  061424  051040  040505  020104
```

```
11906  061432  042110  000122
11907  061436  044524  042515  026504  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
11908  061444  052517  020124  047117
11909  061452  051440  042505  000113
11910  061460  051104  053111  020105  EM105:  .ASCIZ  /DRIVE SIEZED BY OTHER PORT/
11911  061466  044523  055105  042105
11912  061474  041040  020131  052117
11913  061502  042510  020122  047520
11914  061510  052122     000
11915  061513     104  052101  020101  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
11916  061520  044515  041523  050115
11917  061526  020122  044127  046111
11918  061534  020105  040502  020111
11919  061542  042523  000124
11920  061546  047516  047040  046505  EM107:  .ASCIZ  /NO NEM ERROR WHEN REF'ING LOC 760000/
11921  061554  042440  051122  051117
11922  061562  053440  042510  020116
11923  061570  042522  023506  047111
11924  061576  020107  047514  020103
11925  061604  033067  030060  030060
11926  061612     000
11927  061613     111  052116  050122  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
11928  061620  020124  044127  047105
11929  061626  041440  052116  046122
11930  061634  020122  047516  020124
11931  061642  042122  000131
11932  061646  047516  040440  052124  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
11933  061654  047047  047440  020116
11934  061662  042523  045505     000
11935  061667     104  044522  042526  EM112:  .ASCIZ  /DRIVE'S CYLINDER INCORRECT/
11936  061674  051447  041440  046131
11937  061702  047111  042504  020122
11938  061710  047111  047503  051122
11939  061716  041505  000124
11940  061722  041101  051117  026524  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
11941  061730  041440  047101  052047
11942  061736  051040  040505  020104
11943  061744  051502  000106
11944  061750  052113  030461  043040  EM114:  .ASCIZ  /KT11 FAILURE/
11945  061756  044501  052514  042522
11946  061764     000
11947  061765     115  046505  050040  EM115:  .ASCIZ  /MEM PARITY ERR/
11948  061772  051101  052111  020131
11949  062000  051105  000122
11950  062004  052503  051122  047105  DH100:  .ASCIZ  /CURRENT COMMAND :/
11951  062012  020124  047503  046515
11952  062020  047101  020104  000072
11953  062026  051120  053105  047511  DH105:  .ASCIZ  /PREVIOUS COMMAND :/
11954  062034  051525  041440  046517
11955  062042  040515  042116  035040
11956  062050     000
11957  062051     122  033113  030461  DH200:  .ASCIZ  /RK611 REGISTERS :/
11958  062056  051040  043505  051511
11959  062064  042524  051522  035040
11960  062072     000
11961  062073     122  046505  044501  DH500:  .ASCIZ  /REMAINING REGISTERS NOT VALID/
```

```
11962  062100  044516  043516  051040
11963  062106  043505  051511  042524
11964  062114  051522  047040  052117
11965  062122  053040  046101  042111
11966  062130     000
11967  062131     124  042510  043040  DH501:  .ASCIZ  /THE FOLLOWING REGISTER DATA MAY BE INCORRECT :/
11968  062136  046117  047514  044527
11969  062144  043516  051040  043505
11970  062152  051511  042524  020122
11971  062160  040504  040524  046440
11972  062166  054501  041040  020105
11973  062174  047111  047503  051122
11974  062202  041505  020124  000072
11975  062210  051105  020122  041520  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT/
11976  062216  020040  051104  053111
11977  062224  020105  020040  046503
11978  062232  042116  020040  020040
11979  062240  054503  047114  051104
11980  062246  020040  051124  041501
11981  062254  020113  020040  042523
11982  062262  052103  051117  020040
11983  062270  042127  041440  052116
11984  062276     000
11985  062277     110  020111  040502  DH102:  .ASCIZ  /HI BA   LO BA/
11986  062304  020040  046040  020117
11987  062312  040502     000
11988  062315     120  042522  027126  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
11989  062322  052440  044516  052502
11990  062330  020123  040515  020120
11991  062336  000072
11992  062340  044510  051040  043505  DH104:  .ASCIZ  /HI REGO LO REGO/
11993  062346  020060  047514  051040
11994  062354  043505  000060
11995  062360  052503  027122  052440  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
11996  062366  044516  052502  020123
11997  062374  040515  020120  000072
11998  062402  045522  051503  020061  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
11999  062410  020040  045522  051503
12000  062416  020062  020040  045522
12001  062424  041504  020040  020040
12002  062432  045522  040504  020040
12003  062440  020040  045522  041527
12004  062446  020122  020040  045522
12005  062454  040502  020040  020040
12006  062462  045522  051501  043117
12007  062470     000
12008  062471     122  042113  020123  DH202:  .ASCIZ  /RKDS    RKER/
12009  062476  020040  051040  042513
12010  062504  000122
12011  062506  030101  020060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
12012  062514  020040  030102  020060
12013  062522  020040  020040  030101
12014  062530  020061  020040  020040
12015  062536  030102  020061  020040
12016  062544  020040  030101  020062
12017  062552  020040  020040  030102
```

```
12018  062560  020062  020040  020040
12019  062566  030101  020063  020040
12020  062574  020040  030102  000063
12021  062602  042510  042101  051105  DH601:  .ASCIZ  /HEADER SHOULD BE:/
12022  062610  051440  047510  046125
12023  062616  020104  042502  000072
12024  062624  042510  042101  051105  DH602:  .ASCIZ  /HEADER OF SECTOR 0 THIS CYLINDER IS:/
12025  062632  047440  020106  042523
12026  062640  052103  051117  030040
12027  062646  052040  044510  020123
12028  062654  054503  044514  042116
12029  062662  051105  044440  035123
12030  062670     000
12031  062671     120  041501  020113  DH604:  .ASCIZ  /PACK ADDRESS OF ERROR(S) :/
12032  062676  042101  051104  051505
12033  062704  020123  043117  042440
12034  062712  051122  051117  051 .0
12035  062720  020051  000072
12036  062724  054503  047114  051104  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
12037  062732  020040  051124  041501
12038  062740  020113  020040  042523
12039  062746  052103  051117     000
12040  062753     103  046131  042116  DH6042: .ASCIZ  /CYLNDR  TRACK/
12041  062760  020122  052040  040522
12042  062766  045503     000
12043  062771     105  041503  042040  DH605:  .ASCIZ  /ECC DATA IS:/
12044  062776  052101  020101  051511
12045  063004  000072
12046  063006  047520  020123  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
12047  063014  020040  040520  020124
12048  063022  020040  020040  047503
12049  063030  051122  041505  040524
12050  063036  046102  037505     000
12051  063043     127  051117  030504  DH606:  .ASCIZ  /WORD1   WORD2   WORD3/
12052  063050  020040  053440  051117
12053  063056  031104  020040  053440
12054  063064  051117  031504     000
12055  063071     102  042101  044040  DH607:  .ASCIZ  /BAD HEADER IS :/
12056  063076  040505  042504  020122
12057  063104  051511  035040     000
12058  063111     127  020104  020043  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
12059  063116  020040  043440  047517
12060  063124  020104  020040  041040
12061  063132  042101  020040  020040
12062  063140  044040  020111  044120
12063  063146  020131  046040  020117
12064  063154  044120  020131  053040
12065  063162  052122  040440  020104
12066  063170  045440  050111  051101
12067  063176  000066
12068  063200  047507  042117  020040  DH702:  .ASCIZ  /GOOD    BAD/
12069  063206  020040  040502  000104
12070  063214  040506  046111  047111  DH703:  .ASCIZ  /FAILING DATA WORD :/
12071  063222  020107  040504  040524
12072  063230  053440  051117  020104
12073  063236  000072
```

```
12074  063240  051123  004460  051123  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
12075  063246  004461  051123  004462
12076  063254  051123  000063
12077  063260  052113  030461  051040  DH705:  .ASCIZ  /KT11 REGS :/
12078  063266  043505  020123  000072
12079  063274  042515  020115  042522  DH706:  .ASCIZ  /MEM REGS :/
12080  063302  051507  035040     000
12081  063307     120  004503  047514  DH707:  .ASCIZ  /PC     LOERAD  HIERAD  MEMSYS/
12082  063314  051105  042101  044011
12083  063322  042511  040522  004504
12084  063330  042515  051515  051531
12085  063336     000
12086  063337     120  004503  051503  DH710:  .ASCIZ  /PC     CSR AD  CSR/
12087  063344  020122  042101  041411
12088  063352  051123     000
12089  063355     103  046131  047111  DH711:  .ASCIZ  /CYLINDERS :/
12090  063362  042504  051522  035040
12091  063370     000
12092  063371     122  042113  020123  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
12093  063376  020040  051040  042513
12094  063404  020122  020040  051040
12095  063412  046513  031122  020040
12096  063420  051040  046513  031522
12097  063426     000
12098  063427     105  051122  051117  DH502:  .ASCIZ  /ERROR TRYING TO GET FAILURE INFO/
12099  063434  052040  054522  047111
12100  063442  020107  047524  043440
12101  063450  052105  043040  044501
12102  063456  052514  042522  044440
12103  063464  043116  000117
12104  063470  042523  047503  042116  DH503:  .ASCIZ  /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
12105  063476  052040  046511  047505
12106  063504  052125  047440  041503
12107  063512  051125  042522  020104
12108  063520  042507  052124  047111
12109  063526  020107  051104  053111
12110  063534  020105  052123  052101
12111  063542  051525     000
12112  063545     116  046525  042502  DH800:  .ASCIZ  /NUMBER OF RETRIES:/
12113  063552  020122  043117  051040
12114  063560  052105  044522  051505
12115  063566  000072
12116                                  .EVEN
12117  063570  001116  005500  005476  DT100:  .WORD   $ERRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
12118  063576  001162  001164  001166
12119  063604  001170
12120  063606  001256  001260          DT102:  .WORD   $REG36,$REG37
12121  063612  001174  001176  001200  DT201:  .WORD   $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
12122  063620  001202  001204  001206
12123  063626  001210
12124  063630  001212  001214          DT202:  .WORD   $REG14,$REG15
12125  063634  001216  001220  001222  DT203:  .WORD   $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
12126  063642  001224  001226  001230
12127  063650  001232  001234
12128  063654  001174  001176  001200  DT601:  .WORD   $REG5,$REG6,$REG7
12129  063662  001202  001204  001206  DT602:  .WORD   $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
```

```
12130  063670  001210  001212  001214
12131  063676  001216
12132  063700  005254  005252          DT103:  .WORD   PRMPHO,PRMPLO
12133
12134  063704  000006                  DF01:   .WORD   6               ;NUMBER OF HEADER LINES
12135  063706     000                          .BYTE   0               ;NUMBER OF COL FOR FIRST HDR
12136  063707     000                          .BYTE   0               ;ALL CHARACTERS OCTAL
12137  063710  062210                          .WORD   DH101           ;SECOND HDR LINE
12138  063712     007     000                  .BYTE   7,0             ;NUM OF COL-ALL OCTAL
12139  063714  062277                          .WORD   DH102
12140  063716     002     000                  .BYTE   2,0
12141  063720  062051                          .WORD   DH200
12142  063722     000     000                  .BYTE   0,0
12143  063724  062402                          .WORD   DH201
12144  063726     007     000                  .BYTE   7,0
12145  063730  062073                          .WORD   DH500
12146  063732     000     000                  .BYTE   0,0
12147
12148  063734  000007                  DF02:   .WORD   7
12149  063736     000     000                  .BYTE   0,0
12150  063740  062210                          .WORD   DH101
12151  063742     007     000                  .BYTE   7,0
12152  063744  062277                          .WORD   DH102
12153  063746     002     000                  .BYTE   2,0
12154  063750  062051                          .WORD   DH200
12155  063752     000     000                  .BYTE   0,0
12156  063754  062402                          .WORD   DH201
12157  063756     007     000                  .BYTE   7,0
12158  063760  062471                          .WORD   DH202
12159  063762     002     000                  .BYTE   2,0
12160  063764  062506                          .WORD   DH203
12161  063766     010     000                  .BYTE   10,0
12162
12163  063770  000010                  DF03:   .WORD   10
12164  063772     000     000                  .BYTE   0,0
12165  063774  062210                          .WORD   DH101
12166  063776     007     000                  .BYTE   7,0
12167  064000  062277                          .WORD   DH102
12168  064002     002     000                  .BYTE   2,0
12169  064004  062051                          .WORD   DH200
12170  064006     000     000                  .BYTE   0,0
12171  064010  062402                          .WORD   DH201
12172  064012     007     000                  .BYTE   7,0
12173  064014  062131                          .WORD   DH501           ;"THE FOLLOWING REG DATA MB INCORRECT"
12174  064016     000     000                  .BYTE   0,0
12175  064020  062471                          .WORD   DH202
12176  064022     002     000                  .BYTE   2,0
12177  064024  062506                          .WORD   DH203
12178  064026     010     000                  .BYTE   10,0
12179
12180  064030  000003                  DF04:   .WORD   3
12181  064032     000     000                  .BYTE   0,0
12182  064034  062210                          .WORD   DH101
12183  064036     007     000                  .BYTE   7,0
12184  064040  062277                          .WORD   DH102
12185  064042     002     000                  .BYTE   2,0
```

```
                12186
12187  064044  000007                  DF05:   .WORD   7               ;OPI, HVRC
12188  064046     000      000                 .BYTE   0,0
12189  064050  062210                          .WORD   DH101
12190  064052     007      000                 .BYTE   7,0
12191  064054  062277                          .WORD   DH102
12192  064056     002      000                 .BYTE   2,0
12193  064060  062602                          .WORD   DH601
12194  064062     000      000                 .BYTE   0,0
12195  064064  063043                          .WORD   DH606
12196  064066     003      000                 .BYTE   3,0
12197  064070  062624                          .WORD   DH602
12198  064072     000      000                 .BYTE   0,0
12199  064074  063043                          .WORD   DH606
12200  064076     003      000                 .BYTE   3,0
12201
12202  064100  000007                  DF07:   .WORD   7
12203  064102     000      000                 .BYTE   0,0
12204  064104  062210                          .WORD   DH101
12205  064106     007      000                 .BYTE   7,0
12206  064110  062277                          .WORD   DH102
12207  064112     002      000                 .BYTE   2,0
12208  064114  062671                          .WORD   DH604
12209  064116     000      000                 .BYTE   0,0
12210  064120  062724                          .WORD   DH6041
12211  064122     003      000                 .BYTE   3,0
12212  064124  062771                          .WORD   DH605
12213  064126     000      000                 .BYTE   0,0
12214  064130  063006                          .WORD   DH6051
12215  064132     003      000                 .BYTE   3,0
12216
12217  064134  000005                  DF10:   .WORD   5
12218  064136     000      000                 .BYTE   0,0
12219  064140  062210                          .WORD   DH101
12220  064142     007      000                 .BYTE   7,0
12221  064144  062277                          .WORD   DH102
12222  064146     002      000                 .BYTE   2,0
12223  064150  062671                          .WORD   DH604
12224  064152     000      000                 .BYTE   0,0
12225  064154  062724                          .WORD   DH6041
12226  064156     003      000                 .BYTE   3,0
12227
12228  064160  000004                  DF11:   .WORD   4
12229  064162     000      000                 .BYTE   0,0
12230  064164  062671                          .WORD   DH604
12231  064166     000      000                 .BYTE   0,0
12232  064170  062724                          .WORD   DH6041
12233  064172     003      000                 .BYTE   3,0
12234  064174  063111                          .WORD   DH701
12235  064176     000      000                 .BYTE   0,0
12236
12237  064200  000006                  DF12:   .WORD   6
12238  064202     000      000                 .BYTE   0,0
12239  064204  062210                          .WORD   DH101
12240  064206     007      000                 .BYTE   7,0
12241  064210  062277                          .WORD   DH102
```

```
12242  064212  002  000                .BYTE   2,0
12243  064214  062051                  .WORD   DH200
12244  064216  000  000                .BYTE   0,0
12245  064220  062402                  .WORD   DH201
12246  064222  007  000                .BYTE   7,0
12247  064224  063371                  .WORD   DH204
12248  064226  004  000                .BYTE   4,0
12249
12250  064230  000006           DF13:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12251  064232  000  000                .BYTE   0,0           ;IN HEADER COMPARE ERROR
12252  064234  062210                  .WORD   DH101         ;AND 2ND LEVEL HEADER
12253  064236  007  000                .BYTE   7,0           ;VRC ERROR
12254  064240  062277                  .WORD   DH102
12255  064242  002  000                .BYTE   2,0
12256  064244  062602                  .WORD   DH601
12257  064246  000  000                .BYTE   0,0
12258  064250  063043                  .WORD   DH606
12259  064252  003  000                .BYTE   3,0
12260  064254  063427                  .WORD   DH502
12261  064256  000  000                .BYTE   0,0
12262
12263  064260  000006           DF14:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12264  064262  000  000                .BYTE   0,0           ;IN OPERATION INCOMPLETE ERROR
12265  064264  062210                  .WORD   DH101
12266  064266  007  000                .BYTE   7,0
12267  064270  062277                  .WORD   DH102
12268  064272  002  000                .BYTE   2,0
12269  064274  062602                  .WORD   DH601
12270  064276  000  000                .BYTE   0,0
12271  064300  063043                  .WORD   DH606
12272  064302  003  000                .BYTE   3,0
12273  064304  063427                  .WORD   DH502
12274  064306  000  000                .BYTE   0,0
12275
12276  064310  000011           DF15:  .WORD   11
12277  064312  000  000                .BYTE   0,0
12278  064314  062210                  .WORD   DH101
12279  064316  007  000                .BYTE   7,0
12280  064320  062277                  .WORD   DH102
12281  064322  002  000                .BYTE   2,0
12282  064324  062051                  .WORD   DH200
12283  064326  000  000                .BYTE   0,0
12284  064330  062402                  .WORD   DH201
12285  064332  007  000                .BYTE   7,0
12286  064334  062471                  .WORD   DH202
12287  064336  002  000                .BYTE   2,0
12288  064340  063470                  .WORD   DH503
12289  064342  000  000                .BYTE   0,0
12290  064344  062131                  .WORD   DH501
12291  064346  000  000                .BYTE   0,0
12292  064350  062506                  .WORD   DH203
12293  064352  010  000                .BYTE   10,0
12294
12295  064354  000011           DF16:  .WORD   11
12296  064356  000  000                .BYTE   0,0
12297  064360  062210                  .WORD   DH101
```

```
12298  064362  007      000                .BYTE   7,0
12299  064364  062277                      .WORD   DH102
12300  064366  002      000                .BYTE   2,0
12301  064370  062051                      .WORD   DH200
12302  064372  000      000                .BYTE   0,0
12303  064374  062402                      .WORD   DH201
12304  064376  007      000                .BYTE   7,0
12305  064400  062471                      .WORD   DH202
12306  064402  002      000                .BYTE   2,0
12307  064404  063427                      .WORD   DH502
12308  064406  000      000                .BYTE   0,0
12309  064410  062131                      .WORD   DH501
12310  064412  000      000                .BYTE   0,0
12311  064414  062506                      .WORD   DH203
12312  064416  010      000                .BYTE   10.0
12313
12314  064420  000005                DF17: .WORD   5
12315  064422  000      000                .BYTE   0,0
12316  064424  062210                      .WORD   DH101
12317  064426  007      000                .BYTE   7,0
12318  064430  062277                      .WORD   DH102
12319  064432  002      000                .BYTE   2,0
12320  064434  063355                      .WORD   DH711
12321  064436  000      000                .BYTE   0,0
12322  064440  063200                      .WORD   DH702
12323  064442  002      000                .BYTE   2,0
12324
12325  064444  000002                DF20: .WORD   2
12326  064446  000      000                .BYTE   0,0
12327  064450  062340                      .WORD   DH104
12328  064452  002      000                .BYTE   2,0
12329
12330  064454  000002                DF21: .WORD   2
12331  064456  000      000                .BYTE   0,0
12332  064460  063006                      .WORD   DH6051
12333  064462  003      000                .BYTE   3,0
12334
12335  064464  000006                DF22: .WORD   6
12336  064466  000      000                .BYTE   0,0
12337  064470  062004                      .WORD   DH100
12338  064472  000      000                .BYTE   0,0
12339  064474  062210                      .WORD   DH101
12340  064476  007      000                .BYTE   7,0
12341  064500  062277                      .WORD   DH102
12342  064502  002      000                .BYTE   2,0
12343  064504  062360                      .WORD   DH106
12344  064506  000      000                .BYTE   0,0
12345  064510  062340                      .WORD   DH104
12346  064512  002      000                .BYTE   2,0
12347
12348  064514  000002                DF23: .WORD   2
12349  064516  000      000                .BYTE   0,0
12350  064520  063545                      .WORD   DH800
12351  064522  001      000                .BYTE   1,0
12352
12353  064524  000001                DF24: .WORD   1
```

```
12354  064526  002     000                    .BYTE   2,0
12355
12356  064530  000000                 DF25:   .WORD   0
12357  064532  007     000                    .BYTE   7,0
12358
12359  064534  000002                 DF26:   .WORD   2
12360  064536  002     000                    .BYTE   2,0
12361  064540  062506                        .WORD   DH203
12362  064542  010     000                    .BYTE   10.0
12363
12364  064544  000005                 DF27:   .WORD   5
12365  064546  000     000                    .BYTE   0,0
12366  064550  062210                        .WORD   DH101
12367  064552  007     000                    .BYTE   7,0
12368  064554  062277                        .WORD   DH102
12369  064556  002     000                    .BYTE   2,0
12370  064560  063214                        .WORD   DH703
12371  064562  000     000                    .BYTE   0,0
12372  064564  063200                        .WORD   DH702
12373  064566  002     000                    .BYTE   2,0
12374
12375  064570  000005                 DF30:   .WORD   5
12376  064572  000     000                    .BYTE   0,0
12377  064574  062210                        .WORD   DH101
12378  064576  007     000                    .BYTE   7,0
12379  064600  062277                        .WORD   DH102
12380  064602  002     000                    .BYTE   2,0
12381  064604  063260                        .WORD   DH705
12382  064606  000     000                    .BYTE   0,0
12383  064610  063240                        .WORD   DH704
12384  064612  004     000                    .BYTE   4,0
12385
12386  064614  000005                 DF31:   .WORD   5
12387  064616  000     000                    .BYTE   0,0
12388  064620  062210                        .WORD   DH101
12389  064622  007     000                    .BYTE   7,0
12390  064624  062277                        .WORD   DH102
12391  064626  002     000                    .BYTE   2,0
12392  064630  063274                        .WORD   DH706
12393  064632  000     000                    .BYTE   0,0
12394  064634  063337                        .WORD   DH710
12395  064636  003     000                    .BYTE   3,0
12396
12397  064640                         RWBUF:          ;READ/WRITE DATA BUF (AT LEAST 512(DEC) WORDS)
12398
12399  064640  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
12400  064646  020040  020040  020040
12401  064654  025052  020052  047516
12402  064662  042524  025040  025052
12403  064670  005015    012
12404  064673    101   046114  042040          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12405  064700  044522  042526  020123
12406  064706  047524  041040  020105
12407  064714  042524  052123  042105
12408  064722  046440  051525  020124
12409  064730  040510  042526  035040
```

```
12410  064736  005015     012
12411  064741     061  020056  020101          .ASCII  /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12412  064746  030062  047440  020122
12413  064754  031062  051440  041505
12414  064762  047524  020122  047506
12415  064770  046522  052101  042524
12416  064776  020104  040503  052122
12417  065004  044522  043504  006505
12418  065012     012
12419  065013     062  020056  042510          .ASCII  /2. HEADS MANUALLY LOADED/<15><12>
12420  065020  042101  020123  040515
12421  065026  052516  046101  054514
12422  065034  046040  040517  042504
12423  065042  006504     012
12424  065045     063  020056  042504          .ASCII  /3. DESIRED PORT SELECTED/<15><12>
12425  065052  044523  042522  020104
12426  065060  047520  052122  051440
12427  065066  046105  041505  042524
12428  065074  006504     012
12429  065077     064  020056  051127          .ASCII  /4. WRITE LOCK DISABLED/<15><12>
12430  065104  052111  020105  047514
12431  065112  047503  042040  051511
12432  065120  041101  042514  006504
12433  065126     012
12434  065127     065  020056  051104          .ASCII  /5. DRIVE READY LIGHT ON/<15><12><12>
12435  065134  053111  020105  042522
12436  065142  042101  020131  044514
12437  065150  044107  020124  047117
12438  065156  005015     012
12439  065161     104  044522  042526          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12440  065166  020123  047516  020124
12441  065174  047524  041040  020105
12442  065202  042524  052123  042105
12443  065210  046440  051525  020124
12444  065216  040510  042526  041040
12445  065224  052117  006510     012
12446  065231     120  051117  051524          .ASCIZ  /PORTS DE-SELECTED./<15><12><12>
12447  065236  042040  026505  042523
12448  065244  042514  052103  042105
12449  065252  006456  005012     000
12450
12451  065257     015  044412  051516  HLPFIL: .ASCII  <15><12>/INSTRUCTIONS FOR USING RK06 HEAD ALIGNMENT AID :/<15><12>
12452  065264  051124  041525  044524
12453  065272  047117  020123  047506
12454  065300  020122  051525  047111
12455  065306  020107  045522  033060
12456  065314  044040  040505  020104
12457  065322  046101  043511  046516
12458  065330  047105  020124  044501
12459  065336  020104  006472     012
12460  065343     052  025052  025052          .ASCII  /**************************************************/<15><12>
12461  065350  025052  025052  025052
12462  065356  025052  025052  025052
12463  065364  025052  025052  025052
12464  065372  025052  025052  025052
12465  065400  025052  025052  025052
```

```
 12466  065406  025052  025052  025052
 12467  065414  025052  025052  006452
 12468  065422     012
 12469  065423     115  052517  052116          .ASCII  /MOUNT AN RK06 ALIGNMENT CARTRIDGE ON THE DESIRED /<15><12>
 12470  065430  040440  020116  045522
 12471  065436  033060  040440  044514
 12472  065444  047107  042515  052116
 12473  065452  041440  051101  051124
 12474  065460  042111  042507  047440
 12475  065466  020116  044124  020105
 12476  065474  042504  044523  042522
 12477  065502  020104  005015
 12478  065506  051104  053111  026105          .ASCII  /DRIVE, AND INSURE THAT THE DRIVE IS WRITE-LOCKED. /<15><12>
 12479  065514  040440  042116  044440
 12480  065522  051516  051125  020105
 12481  065530  044124  052101  052040
 12482  065536  042510  042040  044522
 12483  065544  042526  044440  020123
 12484  065552  051127  052111  026505
 12485  065560  047514  045503  042105
 12486  065566  020056  005015
 12487  065572  047503  047116  041505          .ASCII  /CONNECT THE ALIGNMENT INDICATOR TO THE DESIRED DRIVE,/<15><12>
 12488  065600  020124  044124  020105
 12489  065606  046101  043511  046516
 12490  065614  047105  020124  047111
 12491  065622  044504  040503  047524
 12492  065630  020122  047524  052040
 12493  065636  042510  042040  051505
 12494  065644  051111  042105  042040
 12495  065652  044522  042526  006454
 12496  065660     012
 12497  065661     126  040511  052040          .ASCII  /VIA THE HEAD ALIGNMENT CABLE ONLY, AND CYCLE UP THE/<15><12>
 12498  065666  042510  044040  040505
 12499  065674  020104  046101  043511
 12500  065702  046516  047105  020124
 12501  065710  040503  046102  020105
 12502  065716  047117  054514  020054
 12503  065724  047101  020104  054503
 12504  065732  046103  020105  050125
 12505  065740  052040  042510  005015
 12506  065746  051104  053111  027105          .ASCII  /DRIVE./<15><12>
 12507  065754  005015
 12508  065756  042522  050123  047117          .ASCII  /RESPOND TO ALL REQUESTS FOR PARAMETERS, BY ENTERING /<15><12>
 12509  065764  020104  047524  040440
 12510  065772  046114  051040  050505
 12511  066000  042525  052123  020123
 12512  066006  047506  020122  040520
 12513  066014  040522  042515  042524
 12514  066022  051522  020054  054502
 12515  066030  042440  052116  051105
 12516  066036  047111  020107  005015
 12517  066044  044124  020105  042504          .ASCII  /THE DESIRED PARAMETER VALUE (NO <CR> NEEDED)./<15><12>
 12518  066052  044523  042522  020104
 12519  066060  040520  040522  042515
 12520  066066  042524  020122  040526
 12521  066074  052514  020105  047050
```

```
12522  066102  020117  041474  037122
12523  066110  047040  042505  042504
12524  066116  024504  006456     012
12525  066123     124  042510  042522          .ASCII  /THERE ARE TWO MODES OF OPERATION :  MANUAL MODE /<15><12>
12526  066130  040440  042522  052040
12527  066136  047527  046440  042117
12528  066144  051505  047440  020106
12529  066152  050117  051105  052101
12530  066160  047511  020116  020072
12531  066166  046440  047101  040525
12532  066174  020114  047515  042504
12533  066202  006440     012
12534  066205     101  046114  053517          .ASCII  /ALLOWS SELECTION OF DRIVES AND HEADS BY TTY INPUT,/<15><12>
12535  066212  020123  042523  042514
12536  066220  052103  047511  020116
12537  066226  043117  042040  044522
12538  066234  042526  020123  047101
12539  066242  020104  042510  042101
12540  066250  020123  054502  052040
12541  066256  054524  044440  050116
12542  066264  052125  006454     012
12543  066271     101  042116  040440          .ASCII  /AND AUTO MODE ALLOWS DRIVES AND HEADS TO BE SELECTED/<15><12>
12544  066276  052125  020117  047515
12545  066304  042504  040440  046114
12546  066312  053517  020123  051104
12547  066320  053111  051505  040440
12548  066326  042116  044040  040505
12549  066334  051504  052040  020117
12550  066342  042502  051440  046105
12551  066350  041505  042524  006504
12552  066356     012
12553  066357     102  020131  043117          .ASCII  /BY OFF-ON OPERATION OF DRIVE PORT SELECT SWITCHES./<15><12>
12554  066364  026505  047117  047440
12555  066372  042520  040522  044524
12556  066400  047117  047440  020106
12557  066406  051104  053111  020105
12558  066414  047520  052122  051440
12559  066422  046105  041505  020124
12560  066430  053523  052111  044103
12561  066436  051505  006456     012
12562  066443     111  020116  044505          .ASCII  /IN EITHER MODE, UP TO 5 MINUTES OF SEEK EXERCISES/<15><12>
12563  066450  044124  051105  046440
12564  066456  042117  026105  052440
12565  066464  020120  047524  032440
12566  066472  046440  047111  052125
12567  066500  051505  047440  020106
12568  066506  042523  045505  042440
12569  066514  042530  041522  051511
12570  066522  051505  005015
12571  066526  040515  020131  042502          .ASCII  /MAY BE REQUESTED FOR EACH DRIVE. /<15><12>
12572  066534  051040  050505  042525
12573  066542  052123  042105  043040
12574  066550  051117  042440  041501
12575  066556  020110  051104  053111
12576  066564  027105  006440     012
12577  066571     101  051514  020117          .ASCII  /ALSO IN EITHER MODE, A VERIFY OPTION ALLOWS HEAD/<15><12>
```

```
12578  066576  047111  042440  052111
12579  066604  042510  020122  047515
12580  066612  042504  020054  020101
12581  066620  042526  044522  054506
12582  066626  047440  052120  047511
12583  066634  020116  046101  047514
12584  066642  051527  044040  040505
12585  066650  006504     012
12586  066653     123  046105  041505          .ASCII  /SELECTION WITHOUT THE UNLOADING AND LOADING OF /<15><12>
12587  066660  044524  047117  053440
12588  066666  052111  047510  052125
12589  066674  052040  042510  052440
12590  066702  046116  040517  044504
12591  066710  043516  040440  042116
12592  066716  046040  040517  044504
12593  066724  043516  047440  020106
12594  066732  005015
12595  066734  044124  020105  051104          .ASCII  /THE DRIVE BY THE PROGRAM, WHICH OTHERWISE OCCURS/<15><12>
12596  066742  053111  020105  054502
12597  066750  052040  042510  050040
12598  066756  047522  051107  046501
12599  066764  020054  044127  041511
12600  066772  020110  052117  042510
12601  067000  053522  051511  020105
12602  067006  041517  052503  051522
12603  067014  005015
12604  067016  047524  040440  046114          .ASCII  /TO ALLOW MANIPULATION OF THE ALIGNMENT TOOL./<15><12>
12605  067024  053517  046440  047101
12606  067032  050111  046125  052101
12607  067040  047511  020116  043117
12608  067046  052040  042510  040440
12609  067054  044514  047107  042515
12610  067062  052116  052040  047517
12611  067070  027114  005015
12612  067074  047524  051040  051505          .ASCII  /TO RESTART EITHER MODE, TYPE <↑Z>./<15><12>
12613  067102  040524  052122  042440
12614  067110  052111  042510  020122
12615  067116  047515  042504  020054
12616  067124  054524  042520  036040
12617  067132  055136  027076  005015
12618  067140  047524  051040  051505          .ASCII  /TO RESTART ALIGNMENT AID, TYPE <↑R>./<15><12>
12619  067146  040524  052122  040440
12620  067154  044514  047107  042515
12621  067162  052116  040440  042111
12622  067170  020054  054524  042520
12623  067176  036040  051136  027076
12624  067204  005015
12625  067206  047524  051440  046105          .ASCII  /TO SELECT NEW DRIVES IN MANUAL MODE, TYPE <↑C>./<15><12><12>
12626  067214  041505  020124  042516
12627  067222  020127  051104  053111
12628  067230  051505  044440  020116
12629  067236  040515  052516  046101
12630  067244  046440  042117  026105
12631  067252  052040  050131  020105
12632  067260  057074  037103  006456
12633  067266  005012
```

```
12634  067270  047506  020122  042510          .ASCII  /FOR HEAD ALIGNMENT PROCEDURE, REFER TO FIELD /<15><12>
12635  067276  042101  040440  044514
12636  067304  047107  042515  052116
12637  067312  050040  047522  042503
12638  067320  052504  042522  020054
12639  067326  042522  042506  020122
12640  067334  047524  043040  042511
12641  067342  042114  006440     012
12642  067347     124  051505  020124          .ASCIZ  /TEST BOX (RK06TA, RK06TB) OPERATOR'S MANUAL./<15><12>
12643  067354  047502  020130  051050
12644  067362  030113  052066  026101
12645  067370  051040  030113  052066
12646  067376  024502  047440  042520
12647  067404  040522  047524  023522
12648  067412  020123  040515  052516
12649  067420  046101  006456  000012
12650
12651
12652
12653          000001                  .END
```

```
ABASE = 177440          2257#   2349    2390    4807
ABORT = 000200          4234#   9205    3210    9303    9328    9416    9426    9431    9436    9441    9449    9455    9465
                        9499
ABVMX1  003162          3697#
ABVMX2  003164          3698#
ACDW1 = 000000          2349
ACDW2 = 000000          2349
ACLO  = 000010          3298#   9265
ACPUOP= 000000          2349    2364
ADDW0 = 000000          2349
ADDW1 = 000000          2349
ADDW10= 000000          2349
ADDW11= 000000          2349
ADDW12= 000000          2349
ADDW13= 000000          2349
ADDW14= 000000          2349
ADDW15= 000000          2349
ADDW2 = 000000          2349
ADDW3 = 000000          2349
ADDW4 = 000000          2349
ADDW5 = 000000          2349
ADDW6 = 000000          2349
ADDW7 = 000000          2349
ADDW8 = 000000          2349
ADDW9 = 000000          2349
ADEVCT= 000000          2349    2355
ADEVM = 000377          2258#   2349    2391
AENV  = 000000          2349    2360
AENVM = 000000          2349    2361
AFATAL= 000000          2349    2352
ALDRVS  007724          4328#   4875
ALLDRV  015250          4813    4884    4985#
ALLTRM  044336          9304    9439    9446#
ALNCYL= 000365          4227#   7351
ALNPAK  007451          4296#   7266
ALNSEK  031770          6410    6631    7347#
AMADR1= 000000          2349    2377
AMADR2= 000000          2349    2381
AMADR3= 000000          2349    2384
AMADR4= 000000          2349    2387
AMAMS1= 000000          2349    2371
AMAMS2= 000000          2349    2379
AMAMS3= 000000          2349    2382
AMAMS4= 000000          2349    2385
AMSGAD= 000000          2349    2357
AMSGLG= 000000          2349    2358
AMSGTY= 000000          2349    2351
AMTYP1= 000000          2349    2372
AMTYP2= 000000          2349    2380
AMTYP3= 000000          2349    2383
AMTYP4= 000000          2349    2386
ANYDER= 100000          4240#   7258    8920    9487
APASS = 000000          2349    2354
APRIOR= 000000          2349
APTCSU= 000040         10821   11481#
APTENV= 000001         10814   11260   11437   11479#
```

```
APTSIZ= 000200          4754   6259  11478*
APTSPO= 000100         10816  11439  11480*
ASKDRV  024410          6324*  6339   6343   6358   6390   6460
ASKHED  024630          6376*  6404   6416   6423
ASKMDE  016156          5155*  5161   5162   5167   5182   5187   5192   5217   5231   5242   5251   5267   5286
                        5295   5321   5344   5416   7696
ASKMOD  024322          6293   6298   6303*  6317
ASKTMO  015412          5037*  5040   5041   5046   5070   5078   5084   5100   5111   5120   5145
ASKTST  015406          4951   5036*
ASTART  023634          2220   6219*
ASWREG= 000000          2349   2362
AS2SP2  056704          8967  11649*
ATESTN= 000000          2349   2353
AUNIT = 000000          2349   2356
AUSWR = 000000          2349   2363
AUTAL   025446          6486   6493   6500*  6581   6602
AUTALN  013060          4641*  6505
AUTEX   026442          6489   6641*  6662   6675
AUTEXR  013176          4656*  6644
AUTO    025332          6315   6473*  6521   6652   6698   6710
AUTODR  013310          4669*  6679*
AUTOEX  013267          4666*  6680
AUTVRF  013116          4646*  6503
AVECT1= 120210          2256*  2349   2388   4806
AVECT2= 000000          2349   2389
A.ABNL  003036          3579*  5584*  5587*  6533*  6562*  6665*  6670*  7154*  7217*  7221*  7296*  7300*  8236*
                        8532*  8549*  8558*  9181*  9447*  9460*  9462*  10382
A.CONT  003040          3580*  10392
A.NORM  003034          3578*  6118*  6167*  7155*  9182*  9448*  10386
BADDRV  007366          4286*  7224*  7225*  7226   7235*  7236*  7237   7243*  7244*  7245   7263*  7264*  7265
BADSEC= 001000          4236*  8145   8368   8821   8829   9393   9492
BAD632  013314          4671*  7260
BAI   = 000020          3257*  10551
BA16  = 000400          3233*
BA17  = 001000          3234*
BDSCHD  037616          8532   8549   8563*
BDSECT  011251          4466*  8582
BDSRCK  040670          8143   8366   8811*  9392
BGNRTY  044512          9105   9458   9464   9485*
BIT0  = 000001          2148*  3230   3247   3273   3295   3443   3587   3680   4892   4990   6956   7735   8019
                        8036   8051   8090   8107   8161   8203   8308   8323   8354   8390   8434   9064   10666
BIT00 = 000001          2138*  2148
BIT01 = 000002          2137*  2147
BIT02 = 000004          2136*  2146
BIT03 = 000010          2135*  2145
BIT04 = 000020          2134*  2144
BIT05 = 000040          2133*  2143
BIT06 = 000100          2132*  2142
BIT07 = 000200          2131*  2141
BIT08 = 000400          2130*  2140   7860   7892
BIT09 = 001000          2129*  2139  11270  11401
BIT1  = 000002          2147*  3248   3275   3444   3588   3681   4228   5561   5609   8278   9068
BIT10 = 002000          2128*  3235   3264   3286   3321   3335   3348   3363   3377   3456   3597   4237   9097
                       11248
BIT11 = 004000          2127*  3236   3265   3287   3306   3322   3336   3349   3364   3378   3457   4238  11408
BIT12 = 010000          2126*  3238   3266   3288   3323   3337   3350   3351   3365   3379   3458   4239   8979
```

```
                        9007
BIT13 = 020000          2125#   3239    3267    3289    3307    3324    3338    3352    3366    3380    3459    7134    7136
                        8042    8045    8116    8119    8315    8316    8425    8428    11255
BIT14 = 040000          2124#   3240    3268    3290    3308    3325    3339    3353    3367    3381    3460    3598    11383
BIT15 = 100000          2123#   3241    3242    3269    3291    3309    3326    3386    3461    3599    4240    5228    7626
                        9100
BIT2  = 000004          2146#   3249    3276    3277    3297    3445    3589    3682    4229    9072
BIT3  = 000010          2145#   3255    3256    3278    3298    3446    3590    3683    4230    9079
BIT4  = 000020          2144#   3250    3257    3279    3299    3300    3315    3357    3371    3447    3591    3684    4231
                        5479    9083
BIT5  = 000040          2143#   3258    3259    3280    3301    3316    3330    3343    3358    3372    3448    3592    4232
                        6745    8944    9087    10463
BIT6  = 000100          2142#   3231    3260    3281    3302    3317    3331    3344    3359    3373    3449    3594    4233
                        5077    5230    5250    6188    6445    6708    6750    6783    8188    8417
BIT7  = 000200          2141#   3232    3261    3282    3303    3304    3318    3332    3345    3360    3374    3451    4234
                        4804    4987    4999    5598    5603
BIT8  = 000400          2140#   3233    3262    3283    3284    3305    3319    3333    3346    3361    3375    3452    3595
                        4235    9091
BIT9  = 001000          2139#   3234    3263    3285    3320    3334    3347    3362    3376    3453    3596    4236    5464
                        6922    9094    11360
BKSLSH  013415          4683#   7492    7500
BLDEXH  045342          9344    9369    9633#
BLWMIN  003160          3696#
BPTVEC= 000014          2155#
BSE   = 000200          3282#   8563    9384    9816    9982    10218
BSERR = 000002          4228#   5997    6120    8743    9386    9396    9485
BSFACT  004222          3707#   8531    8580    8812
BSSOFT  003222          3706#   8543    8612    8817    8819
BUFFO   005262          3714#   4850    4882    4885    4934    4955    4959    4961    4966    4977    4981    5044    5047
                        5050    5053    5088    5090    5121    5124    5126*   5131    5149    5165    5168    5171    5174
                        5177    5271    5273    5305    5322    5348    5355    5365    5367    5370    5375    5384    5422
                        5425    6867    6871    7495    7509*   7514*   7515    7519*   7636    7638    7658    7661    7663*
                        7668    7684
BUFPRT  005472          3717#
B.BA16= 000001          3247#   10545
B.BA17= 000002          3248#   10545
B.CDT = 000004          3249#   10199   10545
B.CFMT= 000020          3250#   5454    5456    5458    5466    9583    9585    9587    9591    10199   10545   10629
CART    011202          4458#   7840
CCLR  = 100000          3242#   10152
CDT   = 002000          3235#   9958    10508   10565   10616   10639
CERR  = 100000          3241#   9353    9378    9854    9913    9972    10154   10205   10520   10576   10664
CFMT  = 010000          3238#   9612    9645    9958    10508   10565   10616   10639
CHGTST  015630          5052    5081#   5085    5092
CHKDRV  031530          6342    6601    6674    7294#
CHKITR  032176          5517    5703    5780    5868    5969    6052    7408#
CHKLST  015342          4857    4940    4983    5000    5003#
CHKPRM  033276          5311    5390    7717#
CKMEM1  036310          5747    8337#
CKMEM2  036322          5832    8341#
CKMO    036330          8340    8343#
CLEAR = 000105          3201#
CLR   = 000040          3258#
CLRPRM  031070          7159    7176#
CMDTO = 000100          3449#   9228    9549    9715
CMPBUF  034732          5922    8077#
```

```
CMSTRT  013522          4708    4714*   9476
CNTRDY  007622          4316*   4936    8249
CNTRLC  013360          4677*   6749    7463
CNTRLG  013404          4681*   6763    7481
CNTRLR  013372          4679*   6759
CNTRLU  013377          4680*   7475
CNTRLZ  013365          4678*   6755    7469
COE   = 001000          3285*   9310    9816    9982    10218
COMMA   013417          4684*   4897
COMSTR  005236          3709*   8885    8900    9504
CONCLR= 000176          3219*  10647
CONERR  041762          3580    9060*
CR    = 000015          2063*  10860   10870
CRLF  = 000200          2064*  10831   10870
CRMPHO  005260          3713*   8151    8380    8895    8897*
CRMPLO  005256          3712*   8152    8381    8896    8898*
CRTSER  033722          5478    7829*
CR2LF   013411          4682*   8966    8978    9002
CS      005670          3802*   5479    5561    5609    8944
CTLOUT  035640          8220*   8471    8856
CTO   = 004000          3236*   9273   10214
CYLLST  003202          3705*   6091    6098    6127
CYLNDR  005504          3722*   5530*   5533*   5555    5557    5646    5648    6093    6438    6685    7888*   7889*   7890
                        7892*
C.INIT  051120          6110    8476    8861    8871   10425*
C.RTRN  052772         10529   10557   10585   10592   10595   10602   10606   10623   10645   10650   10660   10669   10682
                       10686   10691*
C.SPEC  052012         10465   10561*
DCK   = 100000          3291*   5672    8563    9398    9816    9982   10218
DCKERR= 000020          4231*   9400    9485
DCLO  = 000020          3300*
DDISP = 177570          2070*   2287    4742    6247
DDT   = 000400          3305*
DERCNT  003111          3656*   9337    9362    9407    9418
DESL  = 000010          3255*
DFLTST  005632          3779*   3788    5075    5094    5097    5118    5143
DFQUES  010604          4412*   5264
DFSTRT  013454          2214    4702*
DFTEST  010137          4355*   5081
DF01    063704          2414    2420    2426    2432    2474    2582    2852    2888   12134*
DF02    063734          2438    2456    2462    2468    2480    2486    2492    2498    2504    2510    2516    2522    2528
                        2534    2540    2546    2600    2606    2612    2720    2726    2732    2738    2744    2750    2756
                        2762    2768    2774    2816    2834    2858   12148*
DF03    063770          2444    2450    2822    2828   12163*
DF04    064030          2576   12180*
DF05    064044          2552    2558   12187*
DF07    064100          2564   12202*
DF10    064134          2570   12217*
DF11    064160          2870   12228*
DF12    064200          2618    2624    2630    2636    2642    2648    2654    2660   12237*
DF13    064230          2666    2678   12250*
DF14    064260          2672   12263*
DF15    064310          2684   12276*
DF16    064354          2690   12295*
DF17    064420          2708   12314*
DF20    064444          2876   12325*
```

```
DF21    064454          2786  12330#
DF22    064464          2882  12335#
DF23    064514          2792   2798   2804  12348#
DF24    064524          2810  12353#
DF25    064530          2714   8079*  8133*  8347*  8373*  12356#
DF26    064534          2864  12359#
DF27    064544          2840  12364#
DF30    064570          2894   6940*  12375#
DF31    064614          2900   6983*  6984*  12386#
DH100   062004          2412   2418   2424   2430   2436   2442   2448   2454   2460   2466   2472   2478   2484
                        2490   2496   2502   2508   2514   2520   2526   2532   2538   2544   2550   2556   2562
                        2568   2574   2580   2598   2604   2610   2616   2622   2628   2634   2640   2646   2652
                        2658   2664   2670   2676   2682   2688   2706   2718   2724   2730   2736   2742   2748
                        2754   2760   2766   2772   3814   2820   2826   2832   2838   2850   2856   2886   2892
                        2898  11950#  12337
DH101   062210          8983  11975#  12137  12150  12165  12182  12189  12204  12219  12239  12252  12265  12278
                       12297  12316  12339  12366  12377  12388
DH102   062277          8994  11985#  12139  12152  12167  12184  12191  12206  12221  12241  12254  12267  12280
                       12299  12318  12341  12368  12379  12390
DH103   062315          2874  11988#
DH104   062340         11992#  12327  12345
DH105   062026          8981  11953#
DH106   062360         11995#  12343
DH200   062051         11957#  12141  12154  12169  12243  12282  12301
DH201   062402         11998#  12143  12156  12171  12245  12284  12303
DH202   062471         12008#  12158  12175  12286  12305
DH203   062506         12011#  12160  12177  12292  12311  12361
DH204   063371         12092#  12247
DH500   062073         11961#  12145
DH501   062131         11967#  12173  12290  12309
DH502   063427         12098#  12260  12273  12307
DH503   063470         12104#  12288
DH601   062602         12021#  12193  12256  12269
DH602   062624         12024#  12197
DH604   062671         12031#  12208  12223  12230
DH6041  062724          8583   12036#  12210  12225  12232
DH6042  062753          2808   12040#
DH605   062771          2784   12043#  12212
DH6051  063006         12046#  12214  12332
DH606   063043         12051#  12195  12199  12258  12271
DH607   063071         12055#
DH701   063111          8078*   8132*  8346*  8372*  12058#  12234
DH702   063200          2862   12068#  12322  12372
DH703   063214         12070#  12370
DH704   063240          6941*  12074#  12383
DH705   063260         12077#  12381
DH706   063274         12079#  12392
DH707   063307          6983   12081#
DH710   063337         12086#  12394
DH711   063355         12089#  12320
DH800   063545         12112#  12350
DI    = 040000          3240#  10002  10674
DISPLA  001142          2287#   4742*  4750*  6247*  6255*  11247*  11421*
DISPRE  000174          2211#   4750   6255
DLT   = 100000          3269#   9330   9810   9940   9976  10216
DLTFLG  003132          3673#   5991*  5995   6006
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
DMC     =  000040          3316#
DONE       003113          3658#  8472*  8479   8857*  8864   8868*  8874   8919*  9061*  9175*  9253*  9518*  9521*
DOTIM      003123          3666#
DRA     =  000001          3295#  9805   10588
DRCY    =  000200          3304#
DRIEXR     012550          4603#  6432*
DRISEL     013151          4651#  6600
DRIV       011173          4456#  7798
DRIVE      005500          3720#  5006*  5433*  5440*  5441   5444   5472   6066   6105   6116   6133   6336*  6430
                           6531*  6536   6539   6541*  6563*  6568*  6595   6597   6605   6618   6663*  6666   6677
                           6705*  6706   6723   7162   7163   7207   7224   7235   7243   7263   7294   7303   7311
                           7328   9250   12117
DRNAFG     003127          3670#  8858*  8866   8869*  9074   9247*
DRNAVL=    010000          4239#
DRNRDY     011677          4525#  7313*
DROPDR     011103          4446#  8950
DRQT    =  000040          3301#  9320
DRPAR   =  000010          3278#  9260
DRPDRV=    002000          3456#  10450
DRVCAL     040774          5453   5459   5579   5582   5586   5589   5657   6000   6069   6087   6119   6123   6140
                           6168   6384   6389   6394   6439   6447   6454   6459   6464   6520   6527   6538   6580
                           6590   6594   6609   6611   6615   6624   6669   6686   6692   6697   6704   7161   7219
                           7250   7257   7270   7298   7322   7335   7337   7349   7362   7364   7376   7392   7797
                           7839   7925   8239   8280   8283   8534   8551   8739   8854#  9461   9503   9513   9548
                           9590
DRVDSC=    000040          3448#  9428   9890   10056  10118
DRVERS     003120          3663#  4765*  5438*  8943*  8947   8949*
DRVHRD=    000020          3447#  9423   9897   10092  10223
DRVLST     005606          3754#  4893   4912   4924   4971   4979*  4991*  4994*  5004   5015*  5445   6064   6103
                           6131
DRVLUP     025524          6512#  6548   6560   6616   6637
DRVMSK=    000007          3254#  9799   9840   9991   10436
DRVNED     012043          4544#  7305*
DRVNO      011167          4455#  5472*  5473*
DRVPDT=    000004          3445#  9998   10066
DRVPOS=    000002          3444#  9998   10025  10487
DRVSEL     013160          4653#  6599*
DRVSEQ     007350          4283#  4889
DRVSER     033602          5477   6344   7794#
DRVSZC=    010000          3458#  9241   9828   10590
DRVTST     014532          4705   4855   4871#  4877   4878   4879   4887   4944   4945   5039   5067   5083   5110
                           5160   5214   5239   5266   5294   5343   5415   7693   8228
DRVUSE=    000001          3443#  10450
DRY     =  000200          3303#
DR.CLR=    000005          3226#  9878   10041  10083  10107  10221
DR.SEL=    000001          3225#  9948   10027  10342  10349  10356  10363  10567
DRXOP      007673          4323#  7214
DSC     =  040000          3308#
DSWR    =  177570          2069#  2286   4741   6246
DTBAII=    100000          3461#  5982   6085   10450  10549
DTE     =  010000          3288#  5672   8563   9325   10218
DTYE    =  000040          3280#  9288   10218
DT10C      063570          2413   2419   2425   2431   2437   2443   2449   2455   2461   2467   2473   2479   2485
                           2491   2497   2503   2509   2515   2521   2527   2533   2539   2545   2551   2557   2563
                           2569   2575   2581   2599   2605   2611   2617   2623   2629   2635   2641   2647   2653
                           2659   2665   2671   2677   2683   2689   2707   2719   2725   2731   2737   2743   2749
```

```
                        2755    2761    2767    2773    2815    2821    2827    2833    2839    2851    2857    2881    2887
                        2893    2899   12117#
DT102   063606          12120#
DT103   063700           2875   12132#
DT201   063612          12121#
DT202   063630           2863   12124#
DT203   063634          12125#
DT601   063654           2785    2791    2797    2803    2809    2869   12128#
DT602   063662           2713   12129#
DUACES  010417           4388#   4776
DULACS  003126           3669#   4703*   4712*   4774    4949    7410    9243
DUNPAS  023506           5443    6187#
DZR6N   007024           4242#   4767    6265
ECCNC = 000040           4232#   9403
ECCM  = 020000           3324#
ECM   = 000100           3281#   9401    9816    9982   10216
ECOBAD  027322           5071    5218    5243    6316    6338    6367    6403    6466    6494    6529    6661    6699    6797#
                         7373
EMTVEC= 000030           2158#   4725*   4726*   6230*   6231*
EM1     056711           2411   11650#
EM10    057124           2453   11677#
EM100   061235           2801   11884#
EM101   061260           2807   11888#
EM102   061336           2813   11896#
EM103   061410           2819   11903#
EM104   061436           2825   11907#
EM105   061460           2831   11910#
EM106   061513           2837   11915#
EM107   061546           2843   11920#
EM11    057133           2459   11679#
EM110   061613           2849   11927#
EM111   061646           2855   11932#
EM112   061667           2861   11935#
EM113   061722           2885   11940#
EM114   061750           2891   11944#
EM115   061765           2897   11947#
EM12    057146           2465   11681#
EM13    057167           2471   11684#
EM14    057211           2477   11688#
EM15    057237           2483   11692#
EM16    057260           2489   11695#
EM17    057275           2495   11698#
EM2     056735           2417   11654#
EM20    057316           2501   11701#
EM21    057333           2507   11704#
EM22    057353           2513   11707#
EM23    057375           2519   11711#
EM24    057426           2525   11716#
EM25    057446           2531   11719#
EM26    057471           2537   11723#
EM27    057503           2543   11725#
EM3     056761           2423   11658#
EM30    057526           2549    2669   11729#
EM31    057553           2555    2675   11733#
EM32    057574           2561   11736#
EM33    057615           2567   11739#
```

```
EM34    057637          2573    2873   11743#
EM35    057657          2579   11746#
EM36    057713          2585   11751#
EM37    057746          2591   11756#
EM4     057004          2429   11662#
EM40    060011          2597   11762#
EM41    060055          2603   11769#
EM42    060113          2609   11775#
EM43    060143          2615   11780#
EM5     057025          2435   11665#
EM52    060200          2657   11785#
EM53    060226          2663   11789#
EM56    060253          2681    2687   11793#
EM6     057044          2441   11668#
EM60    060272          2693   11796#
EM61    060326          2699   11801#
EM62    060374          2705   11808#
EM63    060420          2621    2717   11812#
EM64    060465          2627    2723   11819#
EM65    060530          2633    2729   11825#
EM66    060556          2639    2735   11829#
EM67    060611          2645    2741   11834#
EM7     057070          2447   11672#
EM70    060650          2651    2747   11840#
EM71    060717          2753   11847#
EM72    060757          2759   11853#
EM73    061026          2765   11860#
EM74    061062          2771   11865#
EM75    061102          2777   11868#
EM76    061132          2783   11872#
EM77    061214          2789    2795   11881#
ENBCSR  030030          4763    4874    5156   6950#   6996
ENTDRV  011633          4518#   6324
ENTHED  012724          4622#   6376
ENTLCI  010111          4351#   5037
ENTMOD  012416          4586#   6348    6477
ENTPAS  011035          4439#   5413
EQJALS  013442          4693#   7595
ERRCNT  003116          3661#   4704#   4854#   6290#   7389#   8922#   8924    8927    8929#   9062#   9176#   9474#   9494
                        9497
ERRCOM  005476          3719#   9113#   9114#  12117
ERRFRE  041220          3578    6167    6169    7155   8919#   9448
ERRHDL  042466          3579    5587    5674    6562    6670    6718    7154    7221    7300    8236    8558    8565   9174#
                        9447    9462
ERRLMT  003117          3662#   9494
ERRVEC= 000004          2151#   4739    4740#   4751#   4759#   4760#   6244    6245#   6256#   6935#   6936#   6951    6952
                        6953#   6962#   6963#   6973    6974    6977#   6978#   6993#   6994#   7000#   8475#   8490#  11388
                       11389#  11391#  11394#  11502   11503   11507#  11512#  11524#  11529#  11538#  11548#  11549#
ER2ENT  042506          9178#   9240    9358    9383    9467
EXPLAN  010243          4368#   5154
E.CCLR= 000001          3587#  10157
E.CERR= 001000          3596#  10231
E.CLAT= 000020          3591#   9883   10047   10088   10112
E.CMTO= 040000          3598#
E.CONT  003042          3581#   9064    9068    9072    9079    9083    9087    9091    9094    9097    9100    9104#   9782#
                        9820#   9883#   9893#   9964#   9986#  10009#  10015#  10047#  10088#  10112#  10157#  10209#  10231#
```

```
                    10370*  10604*  10658*  10666*  10679*
E.DLT  = 000400      3595*   9964
E.DPAR= 002000       3597*   9893   10370
E.ILLD= 000100       3594*  10658
E.MDS  = 100000      3599*   9782   10209   10604
E.NOAT= 000002       3588*  10009
E.SCLR= 000040       3592*  10679
E.UATT= 000004       3589*  10015
E.UDAT= 000010       3590*   9820    9986
E.UNLD= 020000       3459*   9901   10021   10096   10106
FACTBS  011224       4462*   8581
FC      005646       3793*   5527    5565    5654    5975    6078    6117    6276*   8443    8446
FINAOR  034326       7962*   8138    8361
FINCYL  005564       3741*   7992*   8140    8165    8171    8363    8394    8400
FINHEM  040014       8624*   8674
FINSEC  005567       3743*   7980*   8142    8169    8173    8365    8398    8402
FINTRK  005566       3742*   7990*   8141    8167    8172    8364    8396    8401
FMTE   = 000020      3279*   9292   10218
FNDXDP  030456       5709    5786    5874    7057*
FORHLP  011502       4502*   6294
FORMAT  003115       3660*   5460*   5466*   5660    5979    6081    7164    7831    7972    8526    8546    8651    9583
                     9592
FS      005506       3723*   5461*   5467*   5525*   5528    5558    5566    5649    5658*   5662*   8444
FT      005652       3795*   5188    5564    5976    8445
GETPRM  027504       4823    4828    4837    6853*   6893
GETXDP  030536       4798    5752    5840    5928    7084*   8231
GIVEID  024214       6279*   6309    6330    6355    6385    6455    6484    6528    6649    6693
GNS    = ****** L    2210   11635   11636   11637   11638   11639   11642   11643   11644   11645
GO     = 000001      3230*
GTPKAD  045216       9409    9420    9602*
GTPRMS  040366       8719*   8751
GTSWRG  027614       4792    6889*   7482    8254
GTD     040400       8715    8718    8722*
HALTER  012133       4554*
HCRC   = 000400      3283*
HDR.AD  003060       3621*   9934    9938*  10628*
HDR.CT  003062       3622*   9942*  10631*  10634*
HEADNO  013012       4633*   6414*   6635*
HEDPOS  012653       4615*   6411    6632
HEDSEL  013005       4632*   6415    6636
HIERAD= 177742       2202*   6981
HIGOCT  005470       3716*   5309*   5389*   7566*   7569*   7722    7737    7742    7749
HLPFIL  065257       6299   12451*
HLPGVL  033136       3678*   4766*   6292
HLTPRT  044476       4917    6943    6992    8250    8542    9473*   9500
HLTRQC  007652       4320*   8248
HT     = 000011      2061*  10829   10870
HVRC   = 000400      3284*   5672    8563    9359    9816    9982   10218
HVPCER= 000004       4229*   9361    9485
H.HEAD  003065       3625*
IDAE   = 002000      3286*   9315
IDENT   011435       4495*   6291
IE     = 000100      3231*   4937    6576    7229    7316    9349    9374   10162   10226   10514   10555   10621   10688
ILC    = 000001      3273*   9276   10218
ILF    = 000004      3276*   9284
INCRMA  037306       5750    5828    5834    5926    8497*
```

```
INCRSC  040100          8636   8650*  8678
INITMM  027646          6909*  7022   7111
INITSS  030776          4939   5451   5470   5520   5706   5783   5871   5972   6055   6077   6340   6474   6642
                        7154*  7206   7295   7829   8520
INPUTP  016152          4952   5055   5154*  8244
INTCHR  005522          3729*  5063   5065   5068   5210   5212   5215   5235   5237   5240   5439*  5482*  6305
                        6307   6326   6328   6350   6352   6378   6380   6441   6450   6479   6481   6512   6515
                        6522   6645   6647   6650   6653   6657   6687   6689   6694   6746*  6765   6782*  6797
                        7369   7371   8222   8226   8233*  8242   8246   8252
INTMSK  002072          3630*  9251*  9252   9857   9881   9902   9927   10013  10045  10086  10097  10110  10381*
                        10385* 10389* 10438* 10524
INTR  = 000300          3221*
INTVEC= 000020          2156*  4723*  4724*  6228*  6229*
IR    = 000100          3260*
I.AERR  047502          10033  10039  10057  10066*
I.ATTN  047032          9793   9837   9970*
I.CCLR  050066          9871   9921   10152* 10224  10584  10649
I.CSTS  050520          6160   9864   10299*
I.CST1  050542          9868   9920   10303* 10573
I.DRV   003072          3634*  6135   9251   10438  10439
I.ERR   046254          9865   9869*
I.ERRA  046234          9864*  9922
I.ERRC  046242          9862   9855   9867*  10000
I.HCAL  046500          9848   9911*
I.INTR  045620          4780   6273   9772*
I.ISRL  003064          3624*  9786   9789*  9792*  10163* 10460* 10614* 10687*
I.ISSC  050150          9879   9949   10028  10042  10084  10108  10194* 10343  10350  10357  10364
I.RTRY  046100          9790   9807   9834*
I.STAT  050050          9784   9822   9832   9860   9872   9877   9880   9886   9895   9904   9907   9922   9924
                        9950   9953   9962   9966   9988   10004  10011  10017  10029  10043  10050  10062  10082
                        10085  10090  10099  10102  10109  10114  10123*
I.STAT  050624          9876   10081  10340* 10594
I.STOR  050372          10204  10254*
I.UNLD  047750          10023  10106*
KBDHDL  027142          4777   6270   6738*
KIL_DR  000170          2216*  7089   8450
KIPAR0= 172340          2189*  6911   11488  11516
KIPAR1= 172342          2190*
KIPAR2= 172344          2191*
KIPAR3= 172346          2192*
KIPAR4= 172350          2193*
KIPAR5= 172352          2194*
KIPAR6= 172354          2195*  7119*  7137*  8035*  8044*  8106*  8114   8118*  8307*  8317*  8353*  8375   8427*
KIPAR7= 172356          2196*  11532
KIPDR0= 172300          2178*
KIPDR1= 172302          2179*
KIPDR2= 172304          2180*
KIPDR3= 172306          2181*
KIPDR4= 172310          2182*
KIPDR5= 172312          2183*
KIPDR6= 172314          2184*
KIPDR7= 172316          2185*
KTERHD  027734          4782   6931*
LASTWC  005570          3744*  7963   8137*  8360*
LC      005650          3794*  5530   5531   6277*
LOVEC   003150          3691*
```

```
LDMEM1  036144          5740    8295#
LDMEM2  036156          5744    5820    5824    5919    8299#
LDMO    036164          8298    8301#
LEV2ER= 000400          4235#   9178    9233    9340    9346    9365    9371    9463    9519
LF    = 000012          2062#   10864   10870
LKS     003140          3687#
LODBUF  034504          5915    8010#
LODFLS  015674          4921    5089    5094#
LODFPT  016706          5101    5272    5277#
LODP14  034270          5880    7939#
LODSEC  034162          7901#   7923
LOERAD= 177740          2201#   6980
LOOP1   025750          6550#   6555
LOOP2   025754          6551#   6553
LOWOCT  005466          3715#   5308*   5388*   7565*   7570    7727    7729    7732    7735    7740    7745    7748
LS      005510          3724#   5462*   5468*   5526*   5550    5552    5642    5644    5659*   5663*
LSTCYL= 000632          4225#   6277    7252    7835
LSTMEM  007131          4255#   6836
LSTTRK= 000002          4226#   7253    7836
LSTTST  015512          5049    5058#
LT      005654          3796#   5189
MA      005576          3747#   5726    5727    5733    5734    5805    5806    5812    5813    5894    5895    5901    5902
                        7772    7774    8449*   8452*   8453*   8454*   8455*   8507*   8508*   8509*   8716    8717
MAHILM  005572          3745#   6815*   6816*   6830*   6831*   6837    7057    7061    7063    7770    7771
MANAL   024624          6360    6373#
MANALN  012556          4604#   6374
MANEX   025076          6363    6428#
MANEXR  012471          4594#   6433    6467
MANUAL  024404          6312    6322#
MANVR   025060          6366    6421#
MANVRF  012616          4610#   6422
MAPHOO= 170202          2199#   8897
MAP_OO= 170200          2198#   7039    8898
MAWRDC  010455          4394#   7752
MAXHI1= 000000          4216#
MAXHI2= 000000          4218#
MAXHI3= 000000          4220#
MAXHI4= 000001          4222#
MAXLO1= 062031          4215#
MAXLO2= 017500          4217#
MAXLO3= 112160          4219#
MAXLO4= 022370          4221#
MC_K  = 000400          3319#
MDFLAG  003106          3653#   4702*   4711*   4800    4919    5099    5282    5523    6282*   7212    7412    8224
MDFY15  010754          4430#   7629
MDS   = 001000          3263#   9780    10207   10582
MEMABT  003135          3676#   6990
MEMCSR= 172100          2200#   6954    6960    7001    7009
MEMSYS= 177744          2203#   6982
MEMVEC= 000114          3693#   4761*   4762*
MERD  = 001000          3320#
MESMSK= 000017          3313#
MEWD  = 002000          3321#
MIDXFR  040144          8673#   8746
MIND  = 000200          3318#
MINHI = 000000          4214#
```

```
MINL01= 057467          4213#
MMVEC = 000250          2167#   4782*   4783*
MODP15  032752          5314    5393    7618#
MPERHD  030114          4761    6972#
MSP   = 000100          3317#
MULHDL  023406          6118    6158#
MULSEL  012317          4575#   6575
MULTI1  023474          6075    6121    6130    6171#
MXWRDC  033526          5718    5797    5886    7751    7770#
M.CADD= 017760          3389#
M.CDIF= 017760          3388#
M.DPID  053574         10913#
M.DPIM  053534         10884#
M.DP01  053544         10887#  10896
M.DP02  053564         10891   10895#
M.DP40  053632         10924#  10945
M.DP41  053666         10929   10936#
M.DP42  053676         10935   10940#
M.DP44  053730         10947   10952#
M.DPS0  053742         10922   10956#
M.DRV = 000007          3385#
M.HEAD= 007000          3392#
M.ID  = 000003          3387#
M.PAR = 100000          3386#
M.SECT= 000760          3391#
M.SER = 077770          3390#
NCYL1   005512          3725#
NCYL2   005514          3726#
NED   = 010000          3266#   6716    9212    9222    9795    9873   10216   10586
NEDHDL  027064          6533    6665    6716#   7217    7296
NEM   = 004000          3265#   9207    9810    9976   10216
NEWDRV  017470          5436#   5446    6186    8951
NEWMSG  007337          4281#   6857
NEWON   005530          3732#   6510*   6534*   6545    6547    6558    6668*   6671    6728*   7218*   7222    7297*   7301
NEWPAS  017456          4707    5433#   6212
NGOFST  005546          3738#   5576*   5605*   5634    5638
NMTSTS= 000006          3788#   5518    5665    5704    5781    5869    5970    6053
NOCHK = 000400          3452#  10450   10512   10553   10580   10598   10619   10642   10651   10685
NODRLK  012172          4560#   7330*
NODRTS  007571          4311#   4909
NODSC = 004000          3457#   9433   10038
NOFALS  011271          4469#   4908    5629    5636    8589
NONEXD  012031          4542#   7306
NORTRY  003133          3674#   5992*   7158*   9490
NOTALD  010510          4400#   5186    5191
NOTDFT  016752          5270    5287#
NOTLOK  012160          4558#   7331
NOTMSG  064640          4772   12399#
NOTRDY  011665          4523#   7314
NOTYPE  003137          3679#   4770    4773*
NTREDY  007416          4291#   7238
NULINP  015726          5097    5102#
NXDRIV  007377          4288#   7227
NXF   = 000004          3277#
OCTBIN  053014          5132    5306    5386    5426    6872    7669   10716#
OFFSET= 000115          3205#   5581   10495
```

```
OFINUS  005516          3727#
OFSHED  011300          4474#  5563
OFST   = 000004         3297#
ONLINE  005526          3731#  6509*  6547  6557  6558*
OPCOMP  003112          3657#
OPCONT  003121          3664#
OPI    = 020000         3289#  5672  8563  9334  9816  9982  10218
OPIERR= 000010          4230#  9336  9485
OPNLST  016642          5173   5263#  5268  5275
OR     = 000200         3261#
OREQSW= 000020          3684#
OVLODR  007161          4260#
O.WAIT  003044          3601#  9716  9720*  9830*  9834  9850*  9916*  9929*  9944*  10488*  10548*  10610*  10627*
PACK   = 000103         3200#  7249  7336
PARMDE  010371          4384#  5157
PARM0   002620          3490#  3645  7157  7390  8708  8721  9446  9511  9562  9593
PARM1   002704          3522#  7163*  9180  9546  9580
PART   = 000001         2002#  2014  2020  2042  2051  2215  2219  2234  3705  3745  3769  3780  3795
                        3797   3801  3802  3804  3810  3812  3816  3817  3819  3828  3832  3840  3842
                        3846   3853  3855  3859  3860  3862  4242  4472  4473  4710  4767  4795  5193
                        5483   5486  6218  7070  7089  7276  7850  7897  8210  8278  8289  8835  12397
                       12446  12451
PART1   007121          4253#
PART2   007125          4254#  4769  6267
PAT    = 000020         3315#
PATRN   005534          3734#  5715*  5792*  5881*  5883*  5905  8012  8083  8190  8714
PAT00   006024          3867#  8014  8029  8085  8100
PAT01   006064          3889#
PAT02   006124          3911#
PAT03   006164          3933#
PAT04   006224          3955#
PAT05   006264          3977#
PAT06   006324          3998#
PAT07   006364          4020#
PAT08   006424          4041#
PAT09   006464          4063#
PAT10   006524          4085#
PAT11   006564          4107#
PAT12   006624          4129#
PAT13   006664          4151#  5987  6084
PAT14   006724          4173#  7941
PAT15   006764          4193#  5984  7596  7674*  7680*
PB_KT   003102          3645#  9713  9800  9826  9994  10019  10437*
PBSVAL= 001000          3453#  10375
PCA    = 004000         3322#
PCD    = 010000         3323#
PCLKF   003122          3665#
PCVEC   003152          3692#
PFIFTN  010666          4421#  5233
PGE    = 002000         3264#  9280  10216
PIP    = 020000         3307#
PIRQ   = 177772         2068#
PIRQVE= 000240          2162#
PKRB    003146          3690#
PKS     003142          3688#
PKSB    003144          3689#
```

```
PLOFST  005544          3737*  5575*  5607*  5627   5631
PMA     005602          3749*  5733*  5734*  5812*  5813*  5901*  5902*  7023   7025   8031   8102   8182   8184
                        8302   8344   8411   8413   8688*  8689*  8690*  8691   8692   8712*  8713*  8716*  8717*
PROFLT  005674          3807*  5226   5277   5280   5319   5360
PREPAR  030334          5735   5814   5903   7021*  8697   8755
PREPKB  027302          5060   5207   5219   5244   5521   5707   5784   5872   5973   6056   6304   6325   6349
                        6377   6436   6478   6506   6530   6643   6683   6700   6781*  7368   8255
PRMBUF  013430          4691*  7563*  7564
PRMINP  007216          4265*  4818
PRMLIM  005722          3823*  7727   7729   7737   7740   7742   7745
PRMLST  005646          3792*  5225   5278   5318   5359   7565   7569   7732*  7748*  7749*
PRMNEM  005776          3850*  5220   5299   5355   7563   7567   7623   7720
PRMPHO  005254          3711*  8895*  12132
PRMPLO  005252          3710*  8896*  12132
PRMPSP  013451          4695*  5340
PROMPT  013446          4694*  4941   5107   5158   5291   7646
PRVCMD  005222          3708*  4786   6285   8884   9147
PR0     = 000000        2085*  4785   5436   6275   7393   9512   9589
PR1     = 000040        2086*
PR2     = 000100        2087*
PR3     = 000140        2088*
PR4     = 000200        2089*  4778   6271
PR5     = 000240        2090*  3577
PR6     = 000300        2091*
PR7     = 000340        2092*  4714   4762   4783   6219   6274   6936   6978   11340  11341
PS      = 177776        2065*  2066   4714*  4785*  5436*  6219*  6275*  7393*  9702   9708*  9722*  10431  10432*
                        10691*
PSTART  013510          2218   4711*
PSW     = 177776        2066*  9512*  9589*
PT      005666          3801*  5228   5881   5985   7626
PWRMSG  055436          11343  11347*
PWRVEC= 000024          2157*  4729*  4730*  6234*  6235*  11331* 11339* 11340*
P.ASOF= 000032          3474*  9134   10044* 10045  10078* 10309* 10497* 10498
P.A00 = 000040          3477*  7233   7241   7309   7326   9137   9552   10035* 10036  10052* 10053  10366* 10574*
P.A01 = 000044          3479*  6591   6612   7365   9139   10345*
P.A10 = 000050          3481*  9141   10352*
P.A11 = 000054          3483*  7800   9143   10359*
P.BAHI= 000007          3436*  5729*  5808*  5897*  7047*  8694*  9123
P.BALO= 000010          3437*  5726*  5805*  5894*  5990*  6002*  6084*  7046*  7254*  7837*  8274*  8531*  8543*
                        8691*  9126   10531  10628
P.BAR = 000024          3471*  9133   10075* 10305*
P.B00 = 000042          3478*  9138   10030* 10031  10367* 10575*
P.B01 = 000046          3480*  9140   10346*
P.B10 = 000052          3482*  6141   9142   9574   10353*
P.B11 = 000056          3484*  9144   9951*  10360*
P.CMND= 000001          3430*  5452*  5578*  5581*  5583*  5588*  5656*  5741*  5745*  5821*  5825*  5916*  5920*
                        5993*  5999*  6004*  6058*  6086*  6115*  6122*  6126*  6166*  6383*  6388*  6393*  6435*
                        6446*  6453*  6458*  6463*  6519*  6526*  6537*  6579*  6589*  6593*  6608*  6610*  6614*
                        6623*  6667*  6681*  6691*  6696*  6703*  7160*  7216*  7249*  7251*  7269*  7295*  7321*
                        7334*  7336*  7347*  7361*  7363*  7375*  7391*  7796*  7834*  7924*  8238*  8277*  8282*
                        8522*  9114   9459*  9502*  9547*  9588*  9846   10463  10467  10479  10489  10495  10501
                        10504  10510  10535  10542  10561  10608  10625  10647  10656
P.CS1 = 000016          3468*  6161*  7179   9127   9255   9273   9612   9645   9867*  9919*  9957*  9958*  9960*
                        9961   10070* 10072* 10299* 10302* 10453  10507* 10508* 10510* 10514* 10515  10539* 10542*
                        10544* 10545* 10555* 10556  10564* 10565* 10567* 10568  10570* 10571  10576  10615* 10616*
                        10618* 10621* 10622  10638* 10639* 10641* 10644  10663* 10664  10673* 10674
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
P.CS1H= 000007   3435#  5454   5456*  5458*  7164*  7256*  7830*  8521*  9585*  9587*  9591*  9592*  9957
                10198  10507  10544  10564  10615  10629  10638
P.CS2 = 000020   3469#  5669   6716   9128   9202   9207   9212   9214   9220   9222   9280   9330   9414
                 9873  10073* 10303* 10461* 10486  10551* 10552  10563  10582  10586  10612* 10613  10627
P.CYLN= 000002   3431#  5527*  5537*  5540*  5541   5546*  5555   5557*  5565*  5618   5646   5648*  5654*
                 5975*  6061*  6078*  6108*  6117*  6438*  6685*  7252*  7348*  7350*  7351   7835*  7991
                 8443*  8448*  8523*  8625   8679*  9116   9582*  9955  10491  10533  10635
P.DCYL= 000030   3473#  8629   8675   9129   9387   9602   9634  .0077* 10311*
P.DRVN= 000000   3429#  6068*  6107*  6116*  6139*  6514*  6536*  6588*  6605*  6618*  6666*  7162*  7207*
                 7294*  9843  10020  10196  10435
P.DS  = 000036   3476#  9135   9265   9269   9320  10080* 10307* 10588
P.DTS = 000026   3472#  8631   8633   8676   8677   9130   9388   9390   9573   9605   9606   9635   9641
                10076* 10306*
P.EPAT= 000062   3486#  9411   9869*  10455
P.EPOS= 000060   3485#  9410   9870*
P.ER  = 000034   3475#  5672   8563   9136   9201  10079* 10308*
P.OFST= 000006   3434#  5580*  10497
P.PRST= 000014   3439#  5982*  6085*  9228   9241   9423   9428   9433   9438   9549   9715*  9828*  9862*
                 9890*  9897   9901*  9998*  10021  10025* 10038* 10056* 10066* 10092  10096* 10106* 10118*
                10223* 10375* 10450* 10487* 10511* 10512  10537  10543* 10549  10553  10580  10590* 10598
                10619  10642  10651  10685
P.SECT= 000004   3432#  5528*  5550   5552*  5558*  5566*  5623   5642   5644*  5649*  5977*  6079*  7833*
                 7918   7929*  7968   8444*  8528*  8537*  8538   8544*  8554*  8555   8627   8681*  9119
                 9581*  9956  10492  10534  10636
P.TRCK= 000005   3433#  5529*  5543   5547*  5548   5553*  5564*  5612   5651*  5652   5976*  6409*  6630*
                 7253*  7836*  7919*  7921   7926*  7927   7982   8445*  8524*  8626   8680*  9117
P.WC  = 000012   3438#  5721*  5725*  5800*  5804*  5889*  5893*  5978*  5931*  6080*  6083*  7255*  7838*
                 8275*  8499   8530*  8684*  8710   8747   9121  10532
P.WCR = 000022   3470#  9131  10074* 10304*
QNEWSW= 000000   3189
Q.INIT= 040000   3460#
RCDASW= 000010   3683#
RCLREQ= 004000   4238#  9258   9263   9308   9313   9318   9323   9457
RDALHD= 000164   3218#  9846  10625
RDCHR = 104406   6295   7458  11642#
RDCHRS 032272    4876   4928   4943   5038   5082   5109   5159   5265   5293   5342   5414   6858   7452#
                 7630   7648
RDDATA= 000121   3207#  5745   5825   5920   6004   7251   7834   8522
RDGATE= 100000   3326#
RDHDO  045062    9345   9370   9571#
RD4EAD= 000125   3209#  5452   9588   9960  10641
RDSTAT= 000141   3217#  6126   6589   6610   7216   7295   7363   7796   9547  10561
RDY   = 000200   3232#  6158   9453  10518  10571
RECAL = 000113   3204#  5656   6383   6388   6393   6446   6453   6458   6463   6519   6526   6691   6696
                 6703   7269   7334   8238   9459  10504
RECODE 005474    3718#  5997   6120   7258   7388*  8145   8368   8740   8743   8821*  8829*  8920   8930*
                 9177*  9178   9205*  9210*  9231   9233   9258*  9263*  9303*  9308*  9313*  9318*  9323*
                 9328*  9336*  9340   9346   9361*  9365   9371   9386*  9393   9396*  9400*  9403*  9416*
                 9417*  9426*  9431*  9436*  9441*  9449   9455*  9457   9463   9465*  9485   9487*  9492
                 9499*  9519*  9551*
REDBSF 037354    5471   8519#
REFNEM 037204    5994   6005   8470#
REISSU 003130    3671#  9076*
RELEAS= 000140   3216#  10608
REPLPK 007502    4301#  4926
REPSUP 042206    6010   6137   6151   6162   6931   6975   8082   8348   8540   9111#  9183
```

```
RESREG= 104410          6842   6923   7048   7102   7761   7819   7906   7930   7947   7994   8063   8144   8205
                        8284   8324   8367   8435   8510   8615   8640   8698   8726   8822   8907   9047   9162
                        9514   9563   9594   9653  10988  11040  11644#
RESVEC= 000010          2152#
RETANL  044664          9181   9460   9518#
RETNML  044702          5676   6731   8567   9182   9521#
RKASOF= 000016          3185#  9540   9856   9926  10006  10261  10309  10498#  10523
RKBA  = 000004          3180#  9539  10257  10305  10531#
RKBADR  007250          4270#  4822
RKBAS   003026          3575#  4803#  4807#  4821   4824#  7156   9348   9373   9452   9501   9532   9709   9778
                       10442
RKCS1 = 000000          3178#  4937#  6576#  7229#  7316#  9248#  9349#  9353   9374#  9378   9534   9801   9853
                        9911   9961#  9970  10152#  10153  10162#  10201#  10202  10226#  10254  10515#  10517  10556#
                       10568#  10570  10600#  10622#  10644#  10653#  10663  10673  10688#
RKCS2 = 000010          3182#  9535   9779   9939  10197#  10255  10303  10486#  10552#  10563#  10613#  10637#  10662#
RKDA  = 000006          3181#  9537   9956#  10258  10306  10492#  10534#  10636#
RKDB  = 000024          3188#  5463   5464   9350   9351   9352   9375   9376   9377   9935   9936   9937
RKDC  = 000020          3186#  9536
RKDCYL= 000020          3187#  9955#  10262  10311  10491#  10533#  10635#
RKDS  = 000012          3183#  9453   9541   9804  10259  10307
RKECPS= 000030          3193#  9870  10266
RKECPT= 000032          3195#  9869  10267
RKER  = 000014          3184#  9542   9813   9979  10260  10308
RKMR1 = 000026          3189#  9947#  10026#  10263  10340#  10347#  10354#  10361#  10462#
RKMR2 = 000034          3190#  10264  10274
RKMR3 = 000036          3191#  10265  10305
RKPAT = 000032          3194#
RKPOS = 000030          3192#
RKPRI   003032          3577#   4781   4812#  4831   4847#  9708  10432
RKPRTY  007310          4276#   4836
RKVADR  007270          4273#   4827
RKVEC   003030          3576#   4779   4808#  4809#  4826   4829#  6272
RKWC  = 000002          3179#   9538  10256  10304  10532#
RLS   = 000010          3256#  10612
RNDADR  034122          6092    6437   6684   7887#
RNDLNG= 016514          4224#   6434   6659
RNDSHT= 002734          4223#   6655
RUNTST  017362          5193    5411#
RWBUF   064640          3747    5568   5710   5787   5875   6002   6003#  6008   6012   6988   7254   7261   7837
                        7842    7902   7904   8016   8087   8268   8274   8449   8712  12397#
RWNRDY  012755          4627#   7367
R.ABNL  051052          9721    9831   9906   9923  10101  10121  10381#  10591
R.CONT  051076          9783    9821   9885   9894   9965   9987  10010  10016  10049  10089  10113  10158  10210
                       10233   10371  10389#  10605  10659  10668  10681
R.NORM  051064          9859    9952  10061  10385#  10528  10601  10689
SAVPAR  003176          3703#   7035#  8035   8106   8307   8353
SAVPRS  005550          3739#   8709   8720
SAVREG= 104407          6810    6909   7021   7081   7084   7717   7794   7901   7917   7939   7962   8010   8077
                        8139    8266   8295   8299   8337   8341   8362   8497   8577   8624   8673   8707   8719
                        8811    8883   8942   9060   9112   9174   9531   9572   9633  10973  11014  11643#
SAVWRD  003200          3704#   6135   6164#  6165#
SAVXDP  030530          5712    5789   5877   7081#
SCLR  = 000040          3259#  10662
SCNDRV  031116          5007    7206#
SCOPER= 104411          5748    5923  11645#
SCOPE1  055460         11358#  11645
```

```
SCRACH  005532          3733#   4898*   4899*   4900    6511*   6514    6517    6524    6586    6588    6595*   6655*   6659*
                        6682    7493*   7495*   7496    7506*   7507*   7508    8081*   8147    8160*   8163    8345*   8376
                        8389*   8392
SECNL1  010477          4398#   5181
SECNL2  010527          4403#   5185
SEEK   = 000117         3206#   5578    6058    6086    6435    6681    7347    10479   10489
SELDRV= 000101          3199#   6537    6667    10618
SELECT  005524          3730#   5794*   5815    5837*   6508*   6583    6596*
SELP15  010724          4426#   7642
SERNM   011211          4460#   7799    7841
SETPRM  017126          5176    5339#   5345    5350    5394
SETTUP  032130          5516    5702    5738    5779    5867    5913    5968    6051    7386#
SETUPM  037100          5717    5795    5885    8443#
SIZMEM  027334          4795    6810#
SKI    = 000002         3275#   9305
SOFTBS  011237          4464#   8613
SPACE1  013426          4689#   5106    5290    5626    5633    7537    7645
SPACE2  013425          4688#   5622    8594    8600    8989    8993    8996    8999    9019    9024    9039
SPACE3  013424          4687#   5617
SPACE4  013423          4686#
SPACE6  013421          4685#   7544
SFAR   = 020000         3239#   9255    10214
SPDLSS= 000020          3299#   9269
SRHTBS  040526          8772#   8813
SRTSPL= 000111          3203#   6593    6623    7375    10501
SR0    = 177572         2171#   6922*   6932    7126*   7128*   7130*   7132*   8036*   8051*   8107*   8203*   8308*   8323*
                        8354*   8434*   11513   11528*  11535*
SR1    = 177574         2172#   6933
SR2    = 177576         2173#   6934
SR3    = 172516         2174#   5755*   5843*   5931*   6921*   6937    11525*
ST      005672          3803#   5469
STACK  = 001100         2056#   4721    4872    5437    6226    7386*   7387    8237    11342
STALL   034040          7858#   8855
STALLS  005502          3721#   4791*   4873*   5155*   5469*   6284*   7867
STKLMT= 177774          2067#
STPASS  017444          5285    5431#
STRCMD  041110          6109    8473    8859    8883#
SUBCLR= 000177          3220#   5588    6166    6579    6614    7160    7321    7391    9502    10656
SUBMOD  024512          6346#   6368    6395    6449    6465
SUBVER  007044          4245#   4768    6266
SUMHI1  003170          3700#   7754*
SUMHI2  003174          3702#
SUMLO1  003166          3699#   7753*   7755
SUMLO2  003172          3701#
SVAL   = 100000         3309#
SVPRMS  040304          5739    5818    5914    8707#   8752
SWR     001140          2286#   4719    4741*   4743    4749*   4756*   6224    6246*   6248    6254*   6261*   6268*   6269*
                        6889    7860    8161    8188    8278    8390    8417    8952    8979    9007    11248   11255   11267
                        11270   11360   11383   11401   11408
SWREG   000176          2212#   4749    6254    6268    6889    6891    6894*
SWRMSG  007330          4279#   6892
SW0    = 000001         2120#
SW00   = 000001         2110#   2120
SW01   = 000002         2109#   2119
SW02   = 000004         2108#   2118
SW03   = 000010         2107#   2117
```

```
SW04   = 000020          2106*   2116
SW05   = 000040          2105*   2115
SW06   = 000100          2104*   2114
SW07   = 000200          2103*   2113
SW08   = 000400          2102*   2112
SW09   = 001000          2101*   2111
SW1    = 000002          2119*
SW10   = 002000          2100*
SW11   = 004000          2099*
SW12   = 010000          2098*
SW13   = 020000          2097*   8952
SW14   = 040000          2096*
SW15   = 100000          2095*
SW2    = 000004          2118*
SW3    = 000010          2117*
SW4    = 000020          2116*
SW5    = 000040          2115*
SW6    = 000100          2114*
SW7    = 000200          2113*
SW8    = 000400          2112*
SW9    = 001000          2111*
S.ACLO= 000100           3344*
S.BRHM= 000100           3359*
S.BRKE= 040000           3381*
S.CART= 000400           3361*
S.DCLO= 010000           3351*
S.DIB = 002000           3377*
S.DOOR= 000200           3360*
S.DRA = 000040           3330*
S.DROT= 020000           3352*
S.DRY = 000200           3332*   7233    7309
S.DSC = 040000           3339*   9888   10036   10053   10116
S.FLT = 000200           3345*  10031
S.FORM= 001000           3334*
S.FWD = 002000           3363*
S.HDFL= 000200           3374*
S.HDHM= 000040           3358*   6591    6612    7365
S.ICYL= 000040           3343*
S.ILF = 000400           3346*
S.LIMD= 020000           3380*
S.LOAD= 010000           3365*
S.MHD = 000400           3375*
S.NMOV= 010000           3379*
S.OFF = 002000           3335*
S.PAR = 001000           3347*   9891   10368
S.PIP = 020000           3338*   9899   10094
S.PLO = 004000           3378*
S.REV = 004000           3364*
S.RTZ = 020000           3366*
S.SECT= 000020           3371*
S.SKI = 002000           3346*
S.SPIN= 010000           3337*
S.SPLS= 010000           3350*
S.SPOK= 001000           3362*
S.TYPE= 000400           3333*
S.UNLC= 040000           3367*
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
S.JNS = 040000          3353#
S.VV  = 000100          3331#
S.WCLK= 000040          3372#
S.WGAT= 000200          3373#
S.WLE = 004000          3349#
S.WRL = 004000          3336#   7241    7326
S.XOOK= 000020          3357#
S.XERR= 001000          3376#
S0      005656          3797#   5179    5467    5662
S1      005660          3798#   5179    5468    5663
S2      005662          3799#   5183    5461    5658
S3      005664          3800#   5183    5462    5659
TBITVE= 000014          2153#
TCONHI  003156          3695#
TCONLO  003154          3694#
TELDRV  014616          4881    4889#   4946    4957    5001    5014
TKVEC = 000060          2160#   4777*   4778*   6270*   6271*
TLSTHD  010213          4363#   5058    5103
TOPROC  044712          9063    9230    9530#
TPVEC = 000064          2161#
TRACK   005520          3728#   6401*   6409    6412    6604*   6625    6627*   6629*   6630    6633
TRAPVE= 000034          2159#   4727*   4728*   6232*   6233*   11341*
TRKCHK  034204          7917#
TRKNLM  010541          4405#   5190
TRNSFR  040416          5742    5746    5822    5826    5917    5921    8736#
TRTVEC= 000014          2154#
TSTDRN  011146          4452#   5474
TSTING  003110          3655#   4784*   5432*   6089*   6125*   6283*   8220
TSTLST  005616          3768#   3788    5074    5095    5117    5137*   5142    5146*   7421    7545
TSTMOS  010007          4337#   5036
TSTMSG  056672          8973    11647#
TST1    020002          5514#
TST2    021112          5518    5665    5700#
TST3    021422          5704    5777#
TST4    021762          5781    5865#
TST5    022332          5869    5966#
TST6    022636          5970    6049#
TWOTOS= 002000          4237#   9231    9551
TYPALT  013027          4636#   6475
TYPBSF  037642          5481    8577#
TYPOS = 104405          5632    5639    6201    11639#
TYPE  = 104401          4767    4768    4769    4772    4776    4818    4822    4827    4836    4848    4850    4851    4875
                        4885    4886    4889    4897    4900    4908    4909    4918    4926    4936    4941    4955    4956
                        5036    5037    5044    5045    5058    5062    5081    5090    5091    5103    5106    5107    5149
                        5150    5154    5157    5158    5165    5166    5181    5185    5186    5190    5191    5209    5233
                        5246    5264    5273    5274    5290    5291    5322    5323    5340    5348    5349    5413    5422
                        5423    5474    5563    5617    5622    5626    5629    5633    5636    5640    6199    6202    6265
                        6266    6267    6291    6294    6299    6303    6323    6324    6348    6374    6376    6411    6415
                        6422    6433    6448    6467    6475    6477    6503    6505    6575    6600    6632    6636    6644
                        6680    6709    6749    6755    6759    6763    6765    6766    6797    6798    6836    6840    6841
                        6857    6867    6868    6892    7214    7226    7227    7237    7238    7245    7246    7260    7265
                        7266    7306    7314    7331    7367    7463    7469    7475    7481    7492    7496    7500    7508
                        7513    7515    7516    7518    7537    7544    7564    7589    7595    7629    7638    7639    7642
                        7645    7646    7684    7685    7752    7758    7798    7799    7818    7840    7941    7845    8248
                        8249    8581    8582    8583    8584    8589    8590    8594    8600    8604    8613    8950    8966
                        8967    8968    8972    8973    8978    8981    8982    8983    8984    8989    8992    8993    8994
```

```
                        8995    8996    8999    9002    9003    9005    9019    9024    9028    9034    9039    9040    9042
                       10834   11081   11146   11221   11250   11258   11343   11635*
TYPERR  041300          8942*  11257
TYPFMT  003114          3659*
TYPLST  016370          5170    5205*
TYPMAN  011600          4513*   6323
TYPMOD  011537          4507*   6303
TYPOC = 104402          8593    8599    8603    8988    8998    9001    9021   11636*
TYPON = 104404          5625   11638*
TYPOS = 104403          5614    5619    6854    7541    7546    7592    7597    8975   11637*
TYPPAT  032720          5245    7588*   7644
TYPPRM  032640          5208    5289    7563*
TYPTST  032604          5061    5105    7537*
T.ASOF  003004          3562*   6164    9856*   9857    9881    9926*   9927   10006*  10007   10013   10044   10078   10086
                       10110   10261*  10523*  10524
T.BA    002776          3559*   10075   10257*
T.CS1   002770          3554*   6158    6161    9801*   9853*   9854    9867    9878*   9911*   9913    9919    9948*   9970*
                        9972   10002   10027*  10041*  10071*  10072   10083*  10107*  10153*  10154   10194   10198*  10199*
                       10201   10205   10214   10254*  10301*  10302   10342*  10349*  10356*  10363*  10517*  10518   10520
T.CS2   002772          3556*   9779*   9780    9795    9798    9810    9839    9939*   9940    9976    9990    10073   10195*
                       10196*  10197   10207   10216   10255*
T.DA    003000          3560*   10076   10258*
T.DB    003024          3571*
T.DC    003002          3561*   10077   10262*
T.DS    003010          3565*   9804*   9805    10080   10259*
T.ER    003006          3564*   9813*   9816    9979*   9982    10079   10218   10260*
T.MR1   003012          3566*   10263*
T.MR2   003014          3567*   6507*   9888    9899    10035   10052   10094   10116   10264*  10345   10352   10359   10366
T.MR3   003016          3568*   9891    9951    10030   10265*  10346   10353   10360   10367   10368
T.PAT   003022          3570*   10267*
T.POS   003020          3569*   10266*
T.WCR   002774          3558*   10074   10256*
UBMPRS  003134          3675*   5753    5841    5929    6834*   6919    7036    8149    8378    8893
UEXATT= 000010          3446*   9438    9862
UFE   = 000400          3262*   9214    9220    9795    9873   10216   10586
UNLOAD= 000107          3202*   6608    7361   10467
UNLOD   011433          4493*   6603*   6619*
UNS   = 040000          3290*   9300
UPE   = 020000          3267*   9202    9810    9976   10216
VERIFY  011434          4494*   6347*   6406    6421*   6476*   6492*   6501    6606    6621
VFHDSW= 000002          3681*
VV    = 000100          3302*
WAIT4R  032024          6408    6429    7361*
WCOASW= 000004          3682*
WCE   = 040000          3268*   5669    9414    9810    9976   10216
WCEFLG  003131          3672*   5585*   5590    5675*   8533*   8535    8550*   8552    8566*   8738*   8742*
WCERR = 000100          4233*   8740    9417    9485
WC2BIG  010553          4407*
WDSXFR  005562          3740*   8011    8080    8136    8301    8343    8359    8639*   8682    8710*   8711*
WHDSW = 000001          3680*
WLE   = 004000          3287*   9296    9816    9982
WORDSP  013434          4692*   7589
WRCKHD  021056          5584    5669*
WRDATA= 000123          3208*   5741    5821    5916    5993    6115    8277   10539
WRHEAD= 000127          3210*
WR_   = 004000          3306*
```

```
WRTCHK= 000131          3211#   5583    5999    6122    7924    8282   10535
WRTWAT= 040000          3325#
WRTLOK  007432          4293#   7246
WRTSEC  036050          5535    5539    5545    8266#
W.DRV   003104          3651#   9249*   9710*   9825*   9851*   9903*   9918*   9931*   9945*   9993*  10060*  10069*  10098*
                       10120*  10391*  10440*  10503*  10506*  10527*  10579*  10597*
W.MILI  003050          3604#   9705   10671
W.MIN   003056          3620#   9249   10503
W.MTIM  003046          3603#   9703*   9705*
W.SEC   003052          3617#  10440
W.TIME  003070          3626#   9252*   9706    9712*   9824*   9852*   9902*   9917*   9930*   9946*   9992*  10059*  10068*
                       10097*  10119*  10390*  10439*  10526*  10578*  10596*
W.WCK = 000200          3451#  10450   10511   10537   10543
W.WTCH  045470          8478    8863    8873    9698#  10516   10522   10569
W.8SEC  003054          3619#   9903   10098   10506
XOPSAV  005540          3736#   5710*   5711*   5787*   5788*   5875*   5876*   7093    7094    7120
XOPSVD  003125          3668#   4796    4799*   5714*   5791*   5879*   7086    8235*
XOP1    030556          7083    7089#
XOP2    030624          7087    7090    7102#   7109    7140
XOP3    030650          7096    7111#
XOP4    030630          7098    7104#
XCVLAD  003124          3667#   5713*   5790*   5878*   7085*   8229
XXDPAD  005536          3735#   7059*   7063*   7064*   7065*   7092
XXDPCH  003107          3654#
$APTHD  001000          2243    2249#
$ASTAT= ****** U        11459   11474
$ATYC   055766          11430   11432#
$ATY1   055742          11428#
$ATY3   055750          10819   11429#
$ATY4   055760          11263   11431#
$AUTOB  001134          2283#
$BASE   001374          2390#   4803
$BOADR  001122          2278#
$BODAT  001126          2280#
$BELL   001310          2341#   6448    6709   11250   11282
$CHARC  053530         10836*  10846*  10853   10862*  10867#
$CKSWR= ****** U        11642
$CMTAG  001100          2266#   4716    4717    4725    4731    4732    4733    6221    6222    6230    6236    6237    6238
$CM1  = 000040          2298#   2299#   2300#   2301#   2302#   2303#   2304#   2305#   2306#   2307#   2308#   2309#   2310#
                        2311#   2312#   2313#   2314#   2315#   2316#   2317#   2318#   2319#   2320#   2321#   2322#   2323#
                        2324#   2325#   2326#   2327#   2328#   2329#   2330#
$CM2  = 000100          2298#   2299#   2300#   2301#   2302#   2303#   2304#   2305#   2306#   2307#   2308#   2309#   2310#
                        2311#   2312#   2313#   2314#   2315#   2316#   2317#   2318#   2319#   2320#   2321#   2322#   2323#
                        2324#   2325#   2326#   2327#   2328#   2329#   2330#
$CM3  = 000040          2296#   2298
$CM4  = 000011          2330#   2331#   2332#   2333#   2334#   2335#   2336#   2337#   2338#   2339#
$CNTLG  055321         11320#
$CNTLU  055314         11319#
$CORE   056416         11511   11538#
$CPUOP  001346          2364#
$CRLF   001315          2343#   4848    4918    5062    5209    5246    5640    6766    6840    6841    7513    7518    7758
                        7818    7845    8584    8590    8504    8972    8982    8984    8992    8995    9005    9028    9034
                        9042   10835   10870   11258   11282
$CROUT  056446         11538   11545#
$DBLK   054770         11187   11221   11229#
$DB2C   054072         11014#
```

```
$DB20   053752          6838    7571    7756    7843   10973#
$DECVL  054252         11016   11062#
$DEVCT  001330          2355#   5434*   6185*
$DEVM   001376          2391#   4986
$DOAGN  023610          6195    6204    6210#
$DTBL   054760         11190   11225#
$ENDAD  023600          2229    6206#  11277
$ENDCT  023546          4731    6197#   6236
$ENDMG  023617          6199    6214#
$ENULL  023614          6202    6213#
$ENV    001340          2360#  10814   11260   11437   11461
$ENVM   001341          2361#   4754    4804    4987    4999    6259   10816   10821   11439
$EOP    023474          6053    6183#
$EOPCT  023540          4731*   5284*   5428*   6194#   6198    6236*
$ERFLG  001103          2269#  11245*  11282   11358   11372   11397   11399   11405*  11425
$ERMAX  001115          2275#   4734*   4764*   6239*   7409*  11399   11420*  11425
$ERROR  055000          4725    6230   11244#
$ERRPC  001116          2276#  11252*  11253*  11254   11282   12117
$ERRTB  001400          2408#   8963
$ERTTL  001112          2273#  11251*  11282
$ESCAP  001306          2340#   4733*   6238*  11273   11275   11282   11419*
$ETABL  001340          2359#
$ETEND  001400          2255    2392#
$FATAL  001322          2352#  11465*
$FFLG   056206         11428*  11431*  11459   11468*  11476#
$FILLC  001156          2294#  10839   10870
$FILLS  001155          2293#  10870
$GDADR  001120          2277#
$GDDAT  001124          2279#
$GET42  023570          6203#
$GTSWR= ******  J      11641
$HD     = 000000        2031
$HIBTS  001000          2250#
$HINUM  053246          4793*   6280*   7944   10771   10779   10784*  10789#
$HIOCT  053146          5309    5389    6875    7672   10741*  10754#
$ICNT   001104          2270#  11410*  11411   11413*  11424
$INTAG  001135          2284#
$ITEMB  001114          2274#   8958   11254*  11262   11282
$KTNEX  056410         11512   11537#
$KTOUT  056400         11529   11534#
$KT11   056246          5730    5809    5898    6811*   6813    7095    8032    8103    8192    8201    8303    9313    8321
                        8349    8370    8423    8432    8695    8753   11510#  11514*  11537*
$LF     001316          2344#  10870   11282
$LFLG   056205         11469*  11475#
$LONUM  053250          4794*   6281*   7863    7888    7943   10770   10777   10783*  10790#
$LPADR  001106          2271#   4735*   6240*  11345   11403*  11417*  11422   11424
$LPERR  001110          2272#   4736*   5737*   5912*   6241*  11272   11362   11403   11418*  11424
$LSTAD  056512          6815    6828   11550*  11557#
$LSTBK  056514          6819   11551*  11558#
$MADR1  001352          2377#
$MADR2  001356          2381#
$MADR3  001362          2384#
$MADR4  001366          2387#
$MAIL   001320          2251    2255    2350#   4753    5515    5701    5778    5866    5967    6050    6258   10814   11260
                       11416
$MAMSI  001350          2371#
```

```
$MAMS2  001354          2379#
$MAMS3  001360          2382#
$MAMS4  001364          2385#
$MBADR  001002          2251#
$MFLG   056204         11429*  11435   11470*  11474#
$MNEW   055337         11323#
$MSGAD  001334          2357#  11445*  11448
$MSGLG  001336          2358#  11450*
$MSGTY  001320          2351#  11443   11451*  11463   11467*
$MSWR   055326         11321#
$MTYP1  001351          2372#
$MTYP2  001355          2380#
$MTYP3  001361          2383#
$MTYP4  001365          2386#
$MXCNT  055740         11414   11424#
$NULL   001154          2292#  10841   10870
$NWTST= 000001          5486#   5488    5680#   5682    5760#   5762    5848#   5850    5936#   5938    6022#   6024
$OCNT   054550         11118*  11147*  11160#
$OCTVL  054054          7801   10975   11000#
$OMODE  054552         11113*  11117*  11122   11125*  11136*  11162#
$OVER   055724         11384   11404   11412   11421#
$PASS   001326          2354#   4753*   5431*   5475    6191*   6192*   6200    6213    6258*   7414
$PASTM  001006          2253#
$PWRCT  055434         11336*  11337*  11346#
$PWRDN  055350          4729    6234   11331#  11339
$PWRUP  055362         11331   11336#
$QUES   001314          2342#   4851    4886    4956    5045    5091    5150    5166    5274    5323    5349    5423    6798
                        6868    7516    7639    7685   10870   11282
$RAND   053150          7862    7887    7942   10766#
$RDCHR  055174         11298#  11642
$RDDEC= ****** U       11643
$RDLIN= ****** U       11643
$RDOCT= ****** U       11643
$RDSZ = 000000         11319#
$REGAD  001160          2296#
$REG0   001162          2298#   9115   12117
$REG1   001164          2299#  12117
$REG10  001202          2306#   6937*   6982*   8135*   8136*   8137    8175*   8178    8358*   3359*   8360    8404*   8407
                        9350*   9375*   9410*  12121   12129
$REG11  001204          2307#   8176*   8405*   9351*   9376*   9411*  12121   12129
$REG12  001206          2308#   8177*   8406*   9352*   9377*   9404*   9406*  12121   12129
$REG13  001210          2309#   8179*   8181*   8183*   8184*   8408*   8410*   8412*   8413*  12121   12129
$REG14  001212          2310#   6149*   8178*   8180*   8182*   8407*   8409*   8411*  12124   12129
$REG15  001214          2311#   6150*   8185*   8186*   8414*   8415*  12124   12129
$REG16  001216          2312#   8114*   8375*  12125   12129
$REG17  001220          2313#  12125
$REG2   001166          2300#  12117
$REG20  001222          2314#  12125
$REG21  001224          2315#  12125
$REG22  001226          2316#  12125
$REG23  001230          2317#  12125
$REG24  001232          2318#  12125
$REG25  001234          2319#  12125
$REG26  001236          2320#   8986    9146
$REG27  001240          2321#
$REG3   001170          2301#  12117
```

```
$REG30  001242          2322#
$REG31  001244          2323#
$REG32  001246          2324#
$REG33  001250          2325#
$REG34  001252          2326#
$REG35  001254          2327#
$REG36  001256          2328#  9125*  12120
$REG37  001260          2329#  9126*  12120
$REG4   001172          2302#
$REG5   001174          2303#  6011*  6932*  6976*  8151*  8157*  8165   8171*  8380*  8386*  8394   8400*  8926*
                        8927*  9122   9496*  9497*  9533   9602*  9622*  9634*  9648   9651   12121  12128
$REG6   001176          2304#  6012*  6933*  6980*  6988   7004*  8152*  8158*  8167   8172*  8381*  8387*  8396
                        8401*  9603*  9606*  9617   9619*  9621*  9641*  9642*  9643*  9644*  9647*  9649   9650
                       12121  12128
$REG7   001200          2305#  6934*  6981*  6986   7002*  8159*  8169   8173*  8388*  8398   8402*  9604*  9605*
                        9607   9610*  9614*  9616*  9648*  9650*  9652*  12121  12128
$RESRE  056554         11592#  11644
$RTNAD  023612          6212#
$R2A   = ******  U     11645
$SAVRE  056516         11576#  11643
$SCOPE  055504          4723   6229   11380#
$SETUP= 000037          4698#  4722   4723   4725   4727   4729   4731   4732   4733   4735   6189   6227   6228
                        6230   6232   6234   6236   6237   6238   6240   11245  11270  11277  11287  11325  11381
$SIZE   056210          6812   11498#
$SIZEX  056452         11536   11546#
$STUP = 177777          4698#
$SUPRS  054266          6839   7572   7757   7817   7844   11073#
$SVLAD  055670         11392   11415#
$SVPC = 000234          2227#  2232
$SWR  = 167000          2010#  2030   2034   2035   2036   2037   2038   2039   2040   2041   2339   2340   2341
                        4732   4733   4735   4736   5515   5701   5778   5866   5967   6050   6180   6190   6205
                        6211   6213   6237   6238   6240   6241  11236  11237  11238  11239  11240  11248  11255
                       11267  11270  11282  11373  11374  11375  11376  11377  11383  11395  11397  11398  11399
                       11406  11407  11408  11418  11421  11424
$SWREG  001342          2362#  4756   6261
$SWRMK= 000000         11377
$TESTN  001324          2353#  5515*  5701*  5778*  5866*  5967*  6050*  11416#
$TIMES  001304          2339#  4732*  5448*  6190*  6237*  7416*  7421*  8234*  11381  11406*  11411  11414*  11424
$TKB    001146          2289#  6739   6781*  11285  11302  11308
$TKS    001144          2288#  5077*  5230*  5250*  6188*  6445*  6708*  6750*  6783*  11285  11300  11306
$TMP0   001262          2330#
$TMP1   001264          2331#  8776*  8777*  8785
$TMP10  001302          2338#  8789*  8790*  8791
$TMP2   001266          2332#  8772*  8800
$TMP3   001270          2333#  8773*  8801
$TMP4   001272          2334#  8774*  8802
$TMP5   001274          2335#
$TMP6   001276          2336#
$TMP7   001300          2337#
$TN    = 000007         2019#  2030   5486   5515#  5518   5664   5665   5680   5701#  5704   5760   5778#  5781
                        5848   5866#  5869   5936   5967#  5970   6022   6050#  6053
$TNPWR  054202         11021  11022  11042#
$TPB    001152          2291#  10859* 10870
$TPFLG  001157          2295#  10808  10870
$TPS    001150          2290#  10857  10870
$TRAP   056612          4727   6232   11612#
```

```
$TRAP2  056634          11623*  11634
$TRP    = 000012        11627*  11636*  11637*  11638*  11639*  11640*  11642   11643*  11644*  11645*  11646*
$TRPAD  056646          11617   11634*
$TSTM   001004           2252*
$TSTNM  001102           2268*   5447*   6189*   7418    8240*   8974    11247   11282   11372   11415*  11416   11421   11425
$TYPBN= ****** U        11640
$TYPDS  054554          11175*  11639
$TYPE   053252          10808*  11456   11627   11635
$TYPEC  053464          10838   10845   10852   10857*  10858
$TYPEX  053532          10863   10865   10868*
$TYPOC  054352          11116*  11636
$TYPON  054366          11115   11118*  11638
$TYPOS  054326          11111*  11637
$UNIT   001332           2356*   5449*
$UNITM  001010           2254*
$USWR   001344           2363*
$VECT1  001370           2388*   4802
$VECT2  001372           2389*
$XTSTR  055522          11382   11386*
$$GET4= 000000           6205*
$.$.SM= ****** U         3603    9727    9962    10153   10164   10202   10516   10557   10569   10623   10645   10663   10690
                        10699
$DFILL  054551          11112*  11116*  11126   11161*
$$OCAT= ****** U        11257   11383
.       = 067426         2206*   2210*   2215*   2217*   2219*   2227    2228*   2230*   2232*   2233*   2239    2240*   2242*
                         2244*   2265*   2345    3705*   3706*   3707*   3708*   3709*   3714*   3736*   3739*   3749*   4720
                         4735    4736    5737    5912    6213    6217    6225    6240    6241    10870   10942   11000*  11062*
                        11229*  11282   11285   11325   11333   11350*  11424   11425   11477*
.$ASTA= ****** L        11429   11432
.$X     = 001000         2239*   2244
..HIGH  000232           2222*
..LOW   000230           2221*
```

I05


```
B4EOPS   6175#   6184
B4SCOP  11367#  11391
CKEXIT   2906#   8470    8856
COMMEN   2163#
ENDCOM   2163#
ERROR    2057#   6013    6138    6152    6163    6942    6985    7005    8153    8154    8156    8174    8187    8382    836?
         8385    9403    9416    8486    8541    8928    9066    9070    9077    9081    9085    9089    9093    9096    9099
         9102    9103    9204    9209    9216    9218    9224    9226    9235    9237    9239    9245    9257    9262    9267
         9271    9275    9278    9282    9286    9290    9294    9298    9302    9307    9312    9317    9322    9327    9332
         9342    9355    9357    9367    9380    9382    9395    9412    9421    9425    9430    9435    9440    9466    9468
         9498
ESCAPE   2163#
GETPRI   2163#  11506
GETSWR   2163#
MSG      5486#   5488    5680#   5682    5760#   5762    5848#   5850    5936#   5938    6022#   6024
MULT     2163#
NEWTST   2163#   5486    5680    5760    5848    5936    6022
NEXTAD   2905#
NXTAD    2904#
NXTST    2902#   5518    5664    5704    5781    5869    5970    6053
PARMBK      1#   3487    3519
PARMOF      1#   3393
POP      2163#  10785   11216   11471   11472   11597
PUSH     2163#  10766   11175   11432   11434   11455   11577
P.DRVR      1#   3551
REGDEF      1#   3175
REPORT   2163#
R.DRIV      1#   9656
R.QDRV      1#
SCOPE    2058#   5514    5700    5777    5865    5966    6049    6184
SETPRI   2163#
SETTRA  11627#  11636   11637   11638   11639   11642   11643   11644   11645
SETUP    2163#   4715    6220
SKIP     2163#
SLASH    2163#
SPACE    2163#
STARS    2011    2017    2163#   2225    2236    2238    2245    2261    2345    2348    4861    4869    5019    5033    5486
         5513    5680    5699    5760    5776    5848    5864    5936    5965    6022    6048    6177    6734    6736    6771
         6779    6788    6795    6803    6809    6847    6852    6882    6888    6899    6908    6928    6930    6947    6949
         6969    6971    7015    7020    7053    7056    7070    7080    7144    7152    7169    7174    7189    7204    7276
         7292    7344    7346    7357    7360    7381    7385    7398    7429    7450    7527    7535    7552    7561    7577
         7586    7604    7616    7701    7715    7766    7769    7789    7793    7824    7828    7850    7856    7890    7886
         7897    7900    7911    7916    7935    7938    7952    7961    7999    8008    8068    8076    8210    8219    8260
         8265    8289    8294    8329    8336    8440    8442    8460    8469    8494    8496    8515    8518    8571    8576
         8620    8623    8645    8649    8667    8672    8703    8706    8731    8735    8761    8771    8904    8810    8837
         8853    8880    8882    8912    8918    8932    8941    9049    9059    9106    9164    9172    9525    9529    9568
         9570    9599    9601    9627    9632    9665    9696    9730    9770   10133   10150   10168   10192   10239   10252
        10271   10297   10316   10338   10396   10423   10701   10714   10757   10793   10873   10893   10901   10912   10964
        11003   11065   11088   11165   11232   11284   11290   11354   11357   11369   11427   11484   11561   11606
SWRSL    2010#   2163#   4737#   6242#
S.DRVE      1#   3327
TEST     2903#   5486    5680    5760    5848    5936    6022
TRMTRP  11627#
TYPBIN   2163#
TYPDEC   2163#   6200
TYPNAM   2163#
```

```
TYPNUM   2163#
TYPOCS   2163#
TYPOCT   2163#
TYPTXT   2163#
$DECBN      1#
$OCTBN      1#  10699
$$CMRE   2259#   2298   2299   2300   2301   2302   2303   2304   2305   2306   2307   2308   2309   2310   2311
         2312    2313   2314   2315   2316   2317   2318   2319   2320   2321   2322   2323   2324   2325   2326
         2327    2328   2329
$$CMTM   2259#   2330   2331   2332   2333   2334   2335   2336   2337   2338
$$ESCA   2163#
$$NEWT   2163#   5486   5680   5760   5848   5936   6022
$$SET   11627#  11636  11637  11638  11639  11642  11643  11644  11645
$$SETM   4753#   6258#
$$SKIP   2163#
.EQUAT   2010#   2053
.HEADE   2010#   2020
.KT11    2010#   2163
.SETUP   2010#   4698
.SWRHI   2010#   2030
.SWRLO   2010#   2042#  2043   2044
.$ACTI   2010#   2223
.$APTB   2346#
.$APTH   2010#   2234
.$APTY   2010#  11425
.$CATC   2010#   2204
.$CMTA   2010#   2259
.$DB2D   2010#  11001
.$DB20   2010#  10962
.$EOP    2010#   6175
.$ERRO   2010#  11230
.$POWE   2010#
.$RAND   2010#  10755
.$READ   2010#  11282
.$SAVE   2010#  11559
.$SCOP   2010#  11367
.$SIZE   2010#  11482
.$SUPR   2010#  11063
.$TRAP   2010#  11604
.$TYPD   2010#  11163
.$TYPE   2010#  10791
.$TYPO   2010#  11086

. ABS.   067426     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DSKZ:DZR6NC,DSKZ:DZR6NC.SEQ/SOL/CRF/NL:TOC/DOC=DRIVE9.P11/EQ:QNEWSW,DZR6NC.P11
 RUN-TIME: 113 111 10 SECONDS
 RUN-TIME RATIO: 900/236=3.8
 CORE USED:  50K  (99 PAGES)

 DOCUMENT PAGES:  268
```