RK611
RK06, RK07

RK611/06 SS VFY2
CZR6NE0

AH-9142E-MC
FICHE 1 OF 2

RK611
RK06,RK07

RK611/06 SS VFY2
CZR6NE0

AH-9142E-MC
FICHE 2 OF 2

```
                        .REM @
```

IDENTIFICATION

PRODUCT CODE: AC-9140E-MC

PRODUCT NAME: CZR6NE0 RK611/06 SS VFY2

DATE: JUNE 1980

MAINTAINER: DIAGNOSTIC GROUP

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
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	RK06-07 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-RK07 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 RK06-07 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 BEGINNNG OF TESTING, THE FOLLOWING OPTIONS MAY BE SPECIFIED BY THE USER:

- RK611 REGISTER ADDRESS
- RK06-07 VECTOR ADDRESS
- RK06-07 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
- RK06-07 UNIBUS CONTROLLER (RK611)
- 1 TO 8 RK06/07 DRIVES
- 1 TO 8 RK06/07 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-CZR6A THRU CZR6E AND CZR6K) AND RK06 DRIVE DIAGNOSTIC (MAINDEC-11-CZR6H THRU CZR6J) 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).
NOTE: TIMES ARE APPROX DOUBLED FOR RK07 TESTING.

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,RK06/07 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 ERRORS
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: "CZR6ND0 RK11/RK06-RK07 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 DRIVF SELECTION

AFTER AN ADDRESS 204 START, THE PROGRAM INDICATES INPUT MODE BY TYPING "PARAMETER INPUT MODE". THEN, THE RK611 REGISTER ADDRESS, RK06/07 VECTOR ADDRESS AND RK06 PRIORITY ARE OPENED FOR POSSIBLE MODIFICATION AS FOLLOWS:

```
RK06-07 BUS ADR = 177440   NEW=(TYPE NEW VALUE HERE)
RK06-07 VEC ADR =    210   NEW=(TYPE NEW VALUE HERE)
RK06-07 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 D<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) E[illegible]RS EXCEEDED
04	TYPE BAD SECTOR FILES (BS[illegible]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 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-RANDUM	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
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 C, 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 - UFE AND NXD
30. DRIVE ERROR WILL NOT CLEAR
31. DRIVE STATUS CHANGE WILL NOT CLEAR
32. ATTENTION BUT NO STATUS CHANGE OR FAULT
33. ATTENTION BUT DRIVE NOT AVAILABLE
34. ERROR WHILE GATHERING DRIVE STATUS
35. MULTIPLE DRIVE SELECT
36. HEADER COMPARE ERROR
37. ERROR IN RECALIBRATE FOR RECOVERY
38. CLEAR CONTROLLER DID NOT CLEAR ERROR
39. NO ATTENTION IN ATTENTION SUMMARY REGISTER
40. UNSOLICITED ATTENTION
41. UNEXPECTED DATA TYPE ERROR
42. ATTENTION DID NOT RESET WITH CLEAR
43. SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
44. DATA LATE WHEN UNLOADING HEADER
45. CONTROLLER ERROR WHEN DRIVER SERVICING
46. RETRY UNSUCCESSFUL
47. BAD SECTOR ERROR ON SECTOR NOT LISTED BAD

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
```

```
                                        HI BA   LO BA
                                        000000  061566

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

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


                                EXAMPLE 2:

                                      **  DATA MISCOMPARE
                                          TEST 2

                                      PREVIOUS COMMAND:
                                      DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
                                      000000  000131  000016  000001  000000  175000
                                        HI BA   LO BA
                                        000000  074000

                                      CURRENT COMMAND:
                                      ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
                                      033156  000000  000121  000016  000001  000000  175000
                                        HI BA   LO BA
                                        000000  074000

                                      PACK ADDRESS OF ERROR(S):
                                      CYLNDR  TRACK   SECTOR
                                      000016  000001  000000

                                      WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6

                                      000400  000401  125252  000000  075000  155000  000600
                                      000401  000402  125252  000000  075002  155002  000600
                                      000402  000403  125252  000000  075004  155004  000600
                                      000403  000404  125252  000000  075006  155006  000600
                                      000404  000405  125252  000000  075010  155010  000600
                                      000405  000406  125252  000000  075012  155012  000600
                                      000406  000407  125252  000000  075014  155014  000600
                                      000407  000410  125252  000000  075016  155016  000600
                                      000410  000411  125252  000000  075020  155020  000600
                                      000411  000412  125252  000000  075022  155022  000600


                          11.0  RK06 HEAD ALIGNMENT AID

                          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:

```
" *** RK06-07 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 RESPONSE , 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.

```
CZR6ND0 - RK611/RK06-RK07 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.

```
CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION:PART 2


LAST PHYS MEM ADR=377776


PARAMETER INPUT MODE

RK06-07 BUS ADR  = 177440  NEW =
RK06-07 VEC ADR  =    210  NEW =
RK06-07 PRIORITY =      5  NEW =


DRIVE(S) = 0
*

L = LIST TESTS
C = CHANGE TESTS
I = INPUT PARAMETERS AND RUN TES"S

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
```

```
1675
1676
1677
1678                                    5        2 * 0
1679                                    6      100 * 1
1680
1681                                  ENTER L,C, OR I
1682                                  * L
1683
1684                                  TEST    ITERATIONS
1685                                    1        1
1686                                    2        0
1687                                    3        0
1688                                    4        1
1689                                    5        0
1690                                    6        1
1691
1692                                  ENTER L,C, OR I
1693                                  * I
1694
1695                                  T = TYPE PARAMETER LIST
1696                                  O = OPEN PARAMETER LIST
1697                                  S = SET INDIVIDUAL PARAM
1698                                  R = RUN TESTS
1699
1700                                  ENTER T,O,S, OR R
1701                                  * T
1702                                  FC=0
1703                                  LC=632
1704                                  FT=0
1705                                  LT=2
1706                                  S0=0
1707                                  S1=23
1708                                  S2=0
1709                                  S3=25
1710                                  PT=0
1711                                  CS=0
1712                                  ST=0
1713
1714                                  ENTER T,O,S, OR R
1715                                  * S
1716                                  > S2=4
1717                                  > SV=20
1718                                  SV=20?
1719                                  > S3=20
1720                                  > PT=100000
1721                                  TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR>
1722                                  * M
1723
1724                                  MODIFY USER-DEFINED PATTERN 15:
1725                                  WORD 00 = 072307 * 123456!
1726                                  > ^Z
1727
1728                                  ENTER T,O,S, OR R
1729                                  * T
1730                                  FC=0
```

```
1731
1732
1733
1734                                    LC=632
1735                                    FT=0
1736                                    LT=2
1737                                    S0=0
1738                                    S1=23
1739                                    S2=4
1740                                    S3=20
1741                                    PT=100000
1742                                    CS=0
1743                                    ST=0
1744
1745                                    USER-DEFINED PATTERN 15:
1746                                    WORD 00 = 123456
1747                                    WORD 01 = 123456
1748                                    WORD 02 = 123456
1749                                    WORD 03 = 123456
1750                                    WORD 04 = 123456
1751                                    WORD 05 = 123456
1752                                    WORD 06 = 123456
1753                                    WORD 07 = 123456
1754                                    WORD 10 = 123456
1755                                    WORD 11 = 123456
1756                                    WORD 12 = 123456
1757                                    WORD 13 = 123456
1758                                    WORD 14^Z
1759                                     = 123456
1760
1761                                    ENTER T,O,S, OR R
1762                                    * R
1763
1764                                    ENTER NO. OF PASSES (1-77777) :
1765                                    * 1
1766
1767                                    TESTING DRIVE 0
1768                                    DRIVE SER. NO.  8
1769                                    CART. SER. NO.  334
1770
1771
1772                                    OFFSET-TO-FAILURE MEASUREMENTS:
1773                                    TRACK  CYLN  SECT  +OFST  -OFST
1774                                                      (UIN)  (UIN)
1775                                      0     0     4   NONE   NONE
1776                                      0     0    20   NONE   NONE
1777                                      0   631     4   NONE   NONE
1778                                      0   631    20   NONE   NONE
1779                                      1     0     4   NONE   NONE
1780                                      1     0    20   NONE   NONE
1781                                      1   631     4   NONE   NONE
1782                                      1   631    20   NONE   NONE
1783                                      2     0     4   NONE   NONE
1784                                      2     0    20   NONE   NONE
1785                                      2   631     4   NONE   NONE
1786                                      2   631    20   NONE   NONE
```

```
1787
1788
1789
1790
1791                            END PASS #     1
1792                            TO TEST ALL DRIVES TYPE ''A'' <CR>, ELSE <CR>
1793                            *
```

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.

```
CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 2


*** RK06-07 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):
```

```
1850
1851
1852
1853                                    0
1854                                    TYPE <R> WHEN READY:
1855                                    R
1856                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1857                                    HEAD 0 SELECTED
1858
1859                                    ENTER HEAD NO. (0-2):
1860                                    2
1861                                    TYPE <R> WHEN READY:
1862                                    R
1863                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1864                                    HEAD 2 SELECTED
1865
1866                                    ENTER HEAD NO. (0-2):
1867                                    ^Z
1868
1869                                    ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1870                                    E
1871                                    TYPE <R> WHEN READY:
1872                                    R
1873
1874
1875                                    *RANDOM SEEK EXERCISES IN PROGRESS ON DRIVE 0 *
1876                                    ^Z
1877
1878                                    ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1879                                    V
1880
1881
1882                                    * MANUAL SELECT VERIFY *
1883
1884                                    ENTER HEAD NO. (0-2) :
1885                                    2
1886                                    HEADS POSITIONED AT CYLINDER 365 (OCT)
1887                                    HEAD 2 SELECTFD
1888
1889                                    ENTER HEAD NO. (0-2) :
1890                                    ^Z
1891
1892                                    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.

```
CZR6ND0 - RK611/RK06-RK07 SUBSYSTEM VERIFICATION : PART 2


*** RK06-07 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
```

```
1949
1950
1951
1952                            HEADS POSITIONED AT CYLINDER 365 (OCT)
1953                            HEAD 0 SELECTED
1954                            ^Z
1955
1956
1957                            * AUTO SELECT MODE *
1958
1959                            ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1960                            E
1961
1962
1963                            * AUTO SELECT EXERCISES *
1964
1965                            SHORT OR LONG (S OR L) ?
1966                            S
1967                            EXERCISING DRIVE 0
1968
1969
1970                            * AUTO SELECT MODE *
1971
1972                            ALIGN, VERIFY, OR EXERCISE (A, V, OR E)?
1973                            V
1974
1975
1976                            * AUTO SELECT VERIFY *
1977
1978                            DRIVE 0 SELECTED
1979
1980                            HEADS POSITIONED AT CYLINDER 365 (OCT)
1981                            HEAD 0 SELECTED
1982                            HEADS POSITIONED AT CYLINDER 365 (OCT)
1983                            HEAD 1 SELECTED
1984                            HEADS POSITIONED AT CYLINDER 365 (OCT)
1985                            HEAD 2 SELECTED
1986                            ^Z
1987
1988                            ALIGN, VERIFY, OR EXERCISE (A, V, OR E) ?
1989                            @
```

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

                        ;*** REV 006 ***

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

000001                  $TN=1
                        .TITLE  CZR6NE0 RK611/06 SS VFY2
                        ;*COPYRIGHT (C) 1976,1980
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        .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                  CSWR=   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
```

```
2158
2159            177572                  SR0=    177572
2160            177574                  SR1=    177574
2161            177576                  SR2=    177576
2162            172516                  SR3=    172516
2163
2164                                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
2165
2166            172300                  KIPDR0= 172300
2167            172302                  KIPDR1= 172302
2168            172304                  KIPDR2= 172304
2169            172306                  KIPDR3= 172306
2170            172310                  KIPDR4= 172310
2171            172312                  KIPDR5= 172312
2172            172314                  KIPDR6= 172314
2173            172316                  KIPDR7= 172316
2174
2175                                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
2176
2177            172340                  KIPAR0= 172340
2178            172342                  KIPAR1= 172342
2179            172344                  KIPAR2= 172344
2180            172346                  KIPAR3= 172346
2181            172350                  KIPAR4= 172350
2182            172352                  KIPAR5= 172352
2183            172354                  KIPAR6= 172354
2184            172356                  KIPAR7= 172356
2185
2186            170200                  MAPL00=170200
2187            170202                  MAPH00=170202
2188            172100                  MEMCSR=172100                   ;MEMORY CSR REG START ADRS
2189            177740                  LOERAD=177740                   ;11/70 MEM LO ERROR ADRS REG
2190            177742                  HIERAD=177742                   ;11/70 MEM HI ERROR ADRS REG
2191            177744                  MEMSYS=177744                   ;11/70 MEM SYSTEM REG
2192                                    .SBTTL  TRAP CATCHER
2193
2194            000000                          .=0
2195                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
2196                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
2197                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
2198            000174                          .=174
2199  000174   000000                  DISPREG  .WORD  0                ;;SOFTWARE DISPLAY REGISTER
2200  000176   000000                  SWREG:   .WORD  0                ;;SOFTWARE SWITCH REGISTER
2201                                    .SBTTL  STARTING ADDRESS(ES)
2202  000200   000137  013302                  JMP     @#DFSTRT        ;;JUMP TO STARTING ADDRESS OF PROGRAM
2203           000170                  .=170
2204  000170   000000                  KILLDR: .WORD   0                ;IF NOT = 0, IT IS OK TO OVERLAY LOADER
2205           000204                  .=204
2206  000204   000137  013336                  JMP     @#PSTART
2207           000224                  .=224
2208  000224   000137  024074                  JMP     @#ASTART
2209  000230   000700                  ..LOW:  .WORD   700
2210  000232   001100                  ..HIGH: .WORD   1100
2211                                    .SBTTL  ACT11 HOOKS
2212
2213                                    ;;*****************************************************************
```

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

                                ;;**********************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;**********************************************************************
                                        .$X=.   ;;SAVE CURRENT LOCATION
                                        .=24    ;;SET POWER FAIL TO POINT TO ST RT OF PROGRAM
                                        200     ;;FOR APT START UP
                                        .=44    ;;POINT TO APT INDIRECT ADDRES  PNTR.
                                        $APTHDR ;;POINT TO APT HEADER BLOCK
                                        .=.$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)
```

```
2303  001224  000000            $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
2304  001226  000000            $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
2305  001230  000000            $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
2306  001232  000000            $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
2307  001234  000000            $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
2308  001236  000000            $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
2309  001240  000000            $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
2310  001242  000000            $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
2311  001244  000000            $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
2312  001246  000000            $REG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
2313  001250  000000            $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
2314  001252  000000            $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
2315  001254  000000            $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
2316  001256  000000            $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
2317  001260  000000            $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
2318  001262  000000            $TMP0:  .WORD   0               ;;USER DEFINED
2319  001264  000000            $TMP1:  .WORD   0               ;;USER DEFINED
2320  001266  000000            $TMP2:  .WORD   0               ;;USER DEFINED
2321  001270  000000            $TMP3:  .WORD   0               ;;USER DEFINED
2322  001272  000000            $TMP4:  .WORD   0               ;;USER DEFINED
2323  001274  000000            $TMP5:  .WORD   0               ;;USER DEFINED
2324  001276  000000            $TMP6:  .WORD   0               ;;USER DEFINED
2325  001300  000000            $TMP7:  .WORD   0               ;;USER DEFINED
2326  001302  000000            $TMP10: .WORD   0               ;;USER DEFINED
2327  001304  000000            $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
2328  001306  000000            $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
2329  001310  177607  000377    $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
2330  001314     077            $QUES:  .ASCII  /?/             ;;QUESTION MARK
2331  001315     015            $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
2332  001316  000012            $LF:    .ASCIZ  <12>            ;;LINE FEED
2333                            ;;**************************************************************
2334                            .SBTTL  APT MAILBOX-ETABLE
2335
2336                            ;;**************************************************************
2337                            .EVEN
2338  001320                    $MAIL:                          ;;APT MAILBOX
2339  001320  000000            $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
2340  001322  000000            $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
2341  001324  000000            $TESTN: .WORD   ATESTN          ;;TEST NUMBER
2342  001326  000000            $PASS:  .WORD   APASS           ;;PASS COUNT
2343  001330  000000            $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
2344  001332  000000            $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
2345  001334  000000            $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
2346  001336  000000            $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
2347  001340                    $ETABLE:                        ;;APT ENVIRONMENT TABLE
2348  001340     000            $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
2349  001341     000            $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
2350  001342  000000            $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
2351  001344  000000            $USWR:  .WORD   AUSWR           ;;USER SWITCHES
2352  001346  000000            $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
2353                            ;*                              BITS 15-11=CPU TYPE
2354                            ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2355                            ;*                                      11/70=06,PDQ=07,Q=10
2356                            ;*                              BIT 10=REAL TIME CLOCK
2357                            ;*                              BIT  9=FLOATING POINT PROCESSOR
2358                            ;*                              BIT  8=MEMORY MANAGEMENT
```

```
2359  001350  000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
2360  001351  000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
2361                       ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
2362                       ;*                              900 NSEC CORE=001
2363                       ;*                              300 NSEC BIPOLAR=002
2364                       ;*                              500 NSEC MOS=003
2365  001352  000000       $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
2366                       ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
2367  001354  000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
2368  001355  000          $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
2369  001356  000000       $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
2370  001360  000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
2371  001361  000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
2372  001362  000000       $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
2373  001364  000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
2374  001365  000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
2375  001366  000000       $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
2376  001370  120210       $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
2377  001372  000000       $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
2378  001374  177440       $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
2379  001376  000377       $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
2380  001400               $ETEND:
2381                       .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  060353                         EM1
001402  063050                         DH100
001404  064566                         DT100
001406  064702                         DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
001410  060372                         EM2
001412  063050                         DH100
001414  064566                         DT100
001416  064702                         DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
001420  060410                         EM3
001422  063050                         DH100
001424  064566                         DT100
001426  064702                         DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
001430  060426                         EM4
001432  063050                         DH100
001434  064566                         DT100
001436  064702                         DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
001440  060445                         EM5
001442  063050                         DH100
001444  064566                         DT100
001446  064732                         DF02

                                        ;ERROR 6
001450  060464                         EM6                     ;D TO C PARITY ERROR
001452  063050                         DH100
001454  064566                         DT100
001456  064766                         DF03

                                        ;ERROR 7
001460  060503                         EM7                     ;DRIVE DETECTED PARITY ERROR
001462  063050                         DH100
001464  064566                         DT100
```

```
2438  001466  064766                    DF03
2439
2440                                    ;ERROR 10
2441  001470  060522                    EM10                ;AC LOW
2442  001472  063050                    DH100
2443  001474  064566                    DT100
2444  001476  064732                    DF02
2445
2446                                    ;ERROR 11
2447  001500  060531                    EM11                ;SPEED LOSS
2448  001502  063050                    DH100
2449  001504  064566                    DT100
2450  001506  064732                    DF02
2451
2452                                    ;ERROR 12
2453  001510  060544                    EM12                ;ILLEGAL FUNCTION
2454  001512  063050                    DH100
2455  001514  064566                    DT100
2456  001516  064732                    DF02
2457
2458                                    ;ERROR 13
2459  001520  060556                    EM13                ;PROGRAMMING ERROR
2460  001522  063050                    DH100
2461  001524  064566                    DT100
2462  001526  064702                    DF01
2463
2464                                    ;ERROR 14           ;NON-EXISTANT FUNCTION
2465  001530  060567                    EM14
2466  001532  063050                    DH100
2467  001534  064566                    DT100
2468  001536  064732                    DF02
2469
2470                                    ;ERROR 15
2471  001540  060607                    EM15                ;DRIVE TYPE ERROR
2472  001542  063050                    DH100
2473  001544  064566                    DT100
2474  001546  064732                    DF02
2475
2476                                    ;ERROR 16
2477  001550  060623                    EM16                ;FORMAT ERROR
2478  001552  063050                    DH100
2479  001554  064566                    DT100
2480  001556  064732                    DF02
2481
2482                                    ;ERROR 17
2483  001560  060634                    EM17                ;WRITE LOCK ERROR
2484  001562  063050                    DH100
2485  001564  064566                    DT100
2486  001566  064732                    DF02
2487
2488                                    ;ERROR 20
2489  001570  060651                    EM20                ;DRIVE UNSAFE
2490  001572  063050                    DH100
2491  001574  064566                    DT100
2492  001576  064732                    DF02
2493
```

```
2494                                    ;ERROR 21
2495  001600  060664                    EM21                ;SEEK INCOMPLETE
2496  001602  063050                    DH100
2497  001604  064566                    DT100
2498  001606  064732                    DF02
2499
2500                                    ;ERROR 22
2501  001610  060700                    EM22                ;CYLINDER OVERFLOW
2502  001612  063050                    DH100
2503  001614  064566                    DT100
2504  001616  064732                    DF02
2505
2506                                    ;ERROR 23
2507  001620  060713                    EM23                ;ILLEGAL CYLINDER
2508  001622  063050                    DH100
2509  001624  064566                    DT100
2510  001626  064732                    DF02
2511
2512                                    ;ERROR 24
2513  001630  060730                    EM24                ;DRIVE OFF TRACK
2514  001632  063050                    DH100
2515  001634  064566                    DT100
2516  001636  064732                    DF02
2517
2518                                    ;ERROR 25
2519  001640  060746                    EM25                ;DRIVE TIMING ERROR
2520  001642  063050                    DH100
2521  001644  064566                    DT100
2522  001646  064732                    DF02
2523
2524                                    ;ERROR 26
2525  001650  060765                    EM26                ;DATA LATE
2526  001652  063050                    DH100
2527  001654  064566                    DT100
2528  001656  064732                    DF02
2529
2530                                    ;ERROR 27
2531  001660  060777                    EM27                ;CONTROLLER TIMEOUT
2532  001662  063050                    DH100
2533  001664  064566                    DT100
2534  001666  064732                    DF02
2535
2536                                    ;ERROR 30           ;OPERATION INCOMPLETE
2537  001670  061015                    EM30
2538  001672  063050                    DH100
2539  001674  064566                    DT100
2540  001676  065042                    DF05
2541
2542                                    ;ERROR 31
2543  001700  061036                    EM31                ;HEADER VRC ERROR
2544  001702  063050                    DH100
2545  001704  064566                    DT100
2546  001706  065042                    DF05
2547
2548                                    ;ERROR 32
2549  001710  061055                    EM32                ;DATA CHECK ERROR
```

```
2550  001712  063050                  DH100
2551  001714  064566                  DT100
2552  001716  065076                  DF07
2553
2554                                  ;ERROR 33
2555  001720  061074                  EM33                ;WRITE CHECK ERROR
2556  001722  063050                  DH100
2557  001724  064566                  DT100
2558  001726  065132                  DF10
2559
2560                                  ;ERROR 34
2561  001730  061110                  EM34                ;DATA MISCOMPARE(S)
2562  001732  063050                  DH100
2563  001734  064566                  DT100
2564  001736  065026                  DF04
2565
2566                                  ;ERROR 35
2567  001740  061130                  EM35                ;NO DRIVE RESPONSE-UFE & NXD
2568  001742  063050                  DH100
2569  001744  064566                  DT100
2570  001746  064702                  DF01
2571
2572                                  ;ERROR 36
2573  001750  061162                  EM36                ;DRIVE ERROR WILL NOT CLEAR
2574  001752  000000                  0
2575  001754  000000                  0
2576  001756  000000                  0
2577
2578                                  ;ERROR 37           ;DRIVE STATUS CHANGE WILL NOT CLEAR
2579  001760  061211                  EM37
2580  001762  000000                  0
2581  001764  000000                  0
2582  001766  000000                  0
2583
2584                                  ;ERROR 40
2585  001770  061252                  EM40                ;ATTENTION BUT NO STATUS CHANGE OR FAULT
2586  001772  063050                  DH100
2587  001774  064566                  DT100
2588  001776  064732                  DF02
2589
2590                                  ;ERROR 41
2591  002000  061316                  EM41                ;ATTENTION BUT DRIVE NOT AVAILABLE
2592  002002  063050                  DH100
2593  002004  064566                  DT100
2594  002006  064732                  DF02
2595
2596                                  ;ERROR 42
2597  002010  061346                  EM42                ;ATTENTION WHEN NOT EXPECTED
2598  002012  063050                  DH100
2599  002014  064566                  DT100
2600  002016  064732                  DF02
2601
2602                                  ;ERROR 43           ;ERROR WHILE GATHERING DRIVE STATUS
2603  002020  061376                  EM43
2604  002022  063050                  DH100
2605  002024  064566                  DT100
```

```
2606  002026  065176                        DF12
2607
2608                                        ;ERROR 44
2609  002030  061607                        EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2610  002032  063050                        DH100
2611  002034  064566                        DT100
2612  002036  065176                        DF12
2613
2614                                        ;ERROR 45
2615  002040  061645                        EM64                ;NO ATTENTION IN ATTENTION SUMMARY REG
2616  002042  063050                        DH100
2617  002044  064566                        DT100
2618  002046  065176                        DF12
2619
2620                                        ;ERROR 46
2621  002050  061673                        EM65                ;UNSOLICITED ATTENTION
2622  002052  063050                        DH100
2623  002054  064566                        DT100
2624  002056  065176                        DF12
2625
2626                                        ;ERROR 47
2627  002060  061715                        EM66                ;UNEXPECTED DATA TYPE ERROR
2628  002062  063050                        DH100
2629  002064  064566                        DT100
2630  002066  065176                        DF12
2631
2632                                        ;ERROR 50
2633  002070  061741                        EM67                ;ATTENTION DID NOT RESET WITH CLEAR
2634  002072  063050                        DH100
2635  002074  064566                        DT100
2636  002076  065176                        DF12
2637
2638                                        ;ERROR 51
2639  002100  062000                        EM70                ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2640  002102  063050                        DH100
2641  002104  064566                        DT100
2642  002106  065176                        DF12
2643
2644                                        ;ERROR 52
2645  002110  061427                        EM52                ;MULTIPLE DRIVE SELECT
2646  002112  063050                        DH100
2647  002114  064566                        DT100
2648  002116  065176                        DF12
2649
2650                                        ;ERROR 53
2651  002120  061444                        EM53                ;ABREVIATED HCE ERROR
2652  002122  063050                        DH100
2653  002124  064566                        DT100
2654  002126  065226                        DF13
2655
2656                                        ;ERROR 54
2657  002130  061015                        EM30                ;OPERATION INCOMPLETE ERROR
2658  002132  063050                        DH100
2659  002134  064566                        DT100
2660  002136  065256                        DF14
2661
```

```
2662                                    ;ERROR 55
2663  002140  061036                    EM31                ;ABREVIATED HVRC ERROR
2664  002142  063050                    DH100
2665  002144  064566                    DT100
2666  002146  065226                    DF13
2667
2668                                    ;ERROR 56
2669  002150  061467                    EM56                ;2 TIMEOUT ERROR
2670  002152  063050                    DH100
2671  002154  064566                    DT100
2672  002156  065306                    DF15
2673
2674                                    ;ERROR 57           ;2ND LEVEL IN SUBSYSTEM TIMEOUT
2675  002160  061467                    EM56
2676  002162  063050                    DH100
2677  002164  064566                    DT100
2678  002166  065352                    DF16
2679
2680                                    ;ERROR 60
2681  002170  061506                    EM60                ;ERROR IN RECAL FOR RECOVERY
2682  002172  000000                    0
2683  002174  000000                    0
2684  002176  000000                    0
2685
2686                                    ;ERROR 61
2687  002200  061540                    EM61                ;ABORT MESSAGE
2688  002202  000000                    0
2689  002204  000000                    0
2690  002206  000000                    0
2691
2692                                    ;ERROR 62
2693  002210  061573                    EM62                ;CYLINDER MISCOMPARE
2694  002212  063050                    DH100
2695  002214  064566                    DT100
2696  002216  065416                    DF17
2697
2698                                    ;ERROR 63           ;DATA ERROR WORDS
2699  002220  000000                    0
2700  002222  000000                    0
2701  002224  064660                    DT602
2702  002226  065526                    DF25
2703
2704                                    ;ERROR 64
2705  002230  061607                    EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2706  002232  063050                    DH100
2707  002234  064566                    DT100
2708  002236  064732                    DF02
2709
2710                                    ;ERROR 65
2711  002240  061645                    EM64                ;NO ATTENTION IN ATTENTION SUMMARY REG
2712  002242  063050                    DH100
2713  002244  064566                    DT100
2714  002246  064732                    DF02
2715
2716                                    ;ERROR 66
2717  002250  061673                    EM65                ;UNSOLICITED ATTENTION
```

```
2718  002252  063050                    DH100
2719  002254  064566                    DT100
2720  002256  064732                    DF02
2721
2722                                    ;ERROR 67
2723  002260  061715                    EM66                ;UNEXPECTED DATA TYPE ERROR
2724  002262  063050                    DH100
2725  002264  064566                    DT100
2726  002266  064732                    DF02
2727
2728                                    ;ERROR 70
2729  002270  061741                    EM67                ;ATTENTION DID NOT RESET WITH CLEAR
2730  002272  063050                    DH100
2731  002274  064566                    DT100
2732  002276  064732                    DF02
2733
2734                                    ;ERROR 71
2735  002300  062000                    EM70                ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
2736  002302  063050                    DH100
2737  002304  064566                    DT100
2738  002306  064732                    DF02
2739
2740                                    ;ERROR 72
2741  002310  062045                    EM71                ;DATA LATE WHEN UNLOADING HEADER
2742  002312  063050                    DH100
2743  002314  064566                    DT100
2744  002316  064732                    DF02
2745
2746                                    ;ERROR 73
2747  002320  062105                    EM72                ;CONTROLLER ERROR DURING DRIVER SERVICE
2748  002322  063050                    DH100
2749  002324  064566                    DT100
2750  002326  064732                    DF02
2751
2752                                    ;ERROR 74
2753  002330  062140                    EM73                ;DRIVE DETECTED PARITY ERROR
2754  002332  063050                    DH100
2755  002334  064566                    DT100
2756  002336  064732                    DF02
2757
2758                                    ;ERROR 75
2759  002340  062160                    EM74                ;UNDEFINED ERROR
2760  002342  063050                    DH100
2761  002344  064566                    DT100
2762  002346  064732                    DF02
2763
2764                                    ;ERROR 76
2765  002350  062172                    EM75                ;MARKING SECTOR BAD MESSAGE
2766  002352  000000                    0
2767  002354  000000                    0
2768  002356  000000                    0
2769
2770                                    ;ERROR 77
2771  002360  062222                    EM76                ;BAD DATA VERIFICATION WITH READ
2772  002362  063766                    DH605
2773  002364  064652                    DT601
```

```
2774  002366  065452                    DF21
2775
2776                                    ;ERROR 100
2777  002370  062304                    EM77                ;RETRY SUCCESFUL MESSAGE
2778  002372  000000                    0
2779  002374  064652                    DT601
2780  002376  065512                    DF23
2781
2782                                    ;ERROR 101
2783  002400  062304                    EM77                ;ANOTHER RETRY SUCCESSFUL MESSAGE
2784  002402  000000                    0
2785  002404  064652                    DT601
2786  002406  065512                    DF23
2787
2788                                    ;ERROR 102
2789  002410  062325                    EM100               ;RETRY UNSUCCESSFUL MESSAGE
2790  002412  000000                    0
2791  002414  064652                    DT601
2792  002416  065512                    DF23
2793
2794                                    ;ERROR 103
2795  002420  062350                    EM101               ;NO VALID HEADERS IN TRACK JUST READ
2796  002422  063750                    DH6042
2797  002424  064652                    DT601
2798  002426  065522                    DF24
2799
2800                                    ;ERROR 104
2801  002430  062426                    EM102               ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2802  002432  063050                    DH100
2803  002434  064566                    DT100
2804  002436  064732                    DF02
2805
2806                                    ;ERROR  105
2807  002440  062472                    EM103               ;TIMED-OUT ON READ HEADER
2808  002442  063050                    DH100
2809  002444  064566                    DT100
2810  002446  064766                    DF03
2811
2812                                    ;ERROR  106
2813  002450  062520                    EM104               ;TIMED-OUT ON SEEK
2814  002452  063050                    DH100
2815  002454  064566                    DT100
2816  002456  064766                    DF03
2817
2818                                    ;ERROR  107
2819  002460  062542                    EM105               ;DRIVE SIEZED BY OTHER PORT
2820  002462  063050                    DH100
2821  002464  064566                    DT100
2822  002466  064732                    DF02
2823
2824                                    ;ERROR  110
2825  002470  062573                    EM106               ;''DATA MISCMPR WHILE BAI SET''
2826  002472  063050                    DH100
2827  002474  064566                    DT100
2828  002476  065542                    DF27
2829
```

```
2830                                    ;ERROR  111
2831  002500  062626                    EM107                   ;'NO NEM WHEN EXPECTED''
2832  002502  000000                    0
2833  002504  000000                    0
2834  002506  000000                    0
2835
2836                                    ;ERROR  112
2837  002510  062671                    EM110                   ;''INTRPT WHEN CNTRLR NOT READY''
2838  002512  063050                    DH100
2839  002514  064566                    DT100
2840  002516  064702                    DF01
2841
2842                                    ;ERROR  113
2843  002520  062724                    EM111                   ;'NO ATT'N ON SEEK''
2844  002522  063050                    DH100
2845  002524  064566                    DT100
2846  002526  064732                    DF02
2847
2848                                    ;ERROR  114
2849  002530  062745                    EM112                   ;DRIVE'S CYLINDER INCORRECT
2850  002532  064175                    DH702
2851  002534  064626                    DT202
2852  002536  065532                    DF26
2853
2854                                    ;ERROR  115
2855  002540  000000                    0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2856  002542  000000                    0
2857  002544  064652                    DT601
2858  002546  065156                    DF11
2859
2860                                    ;ERROR  116
2861  002550  061110                    EM34                    ;DATA MISCOMPARE (11/70)
2862  002552  063326                    DH103
2863  002554  064676                    DT103
2864  002556  065442                    DF20
2865
2866                                    ;ERROR  117
2867  002560  000000                    0                       ;PART OF DATA MISCOMPARE
2868  002562  000000                    0
2869  002564  064566                    DT100
2870  002566  065462                    DF22
2871
2872                                    ;ERROR  120
2873  002570  062771                    EM113                   ;ABORT- CAN'T READ BSF
2874  002572  063050                    DH100
2875  002574  064566                    DT100
2876  002576  064702                    DF01
2877
2878                                    ;ERROR  121
2879  002600  063017                    EM114                   ;KT11 FAILURE
2880  002602  063050                    DH100
2881  002604  064566                    DT100
2882  002606  065566                    DF30
2883
2884                                    ;ERROR  122
2885  002610  063034                    EM115                   ;MEM PARITY ERROR
```

```
 2886  002612  063050                    DH100
 2887  002614  064566                    DT100
 2888  002616  065612                    DF31
 2889                                    ;ERROR  123
 2890  002620  060410                    EM3                    ;NED IN SIZING UNDER APT
 2891  002622  000000                    0
 2892  002624  000000                    0
 2893  002626  000000                    0
 2894
 2895
 2896
 2897
 2898
 2899
```

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

```
2954                    ;RKDB (DATA BUFFER)
2955                    ; 15     14     13     12     11     10     9      8
2956                    ;-----------------------------------------------------
2957                    ;DB15 ! DB14 ! DB13 ! DB12 ! DB11 ! DB10 ! DB9  ! DB8  !
2958                    ;                    READ/WRITE                       !
2959                    ;-----------------------------------------------------
2960
2961                    ;  7      6      5      4      3      2      1      0
2962                    ;-----------------------------------------------------
2963                    ; DB7 ! DB6  ! DB5  ! DB4  ! DB3  ! DB2  ! DB1  ! DB0  !
2964                    ;                    READ/WRITE                       !
2965                    ;-----------------------------------------------------
2966
2967                    ;RKASOF (ATTENTION SUMMARY AND OFFSET)
2968                    ; 15     14     13     12     11     10     9      8
2969                    ;-----------------------------------------------------
2970                    ;ATN7 ! ATN6 ! ATN5 ! ATN4 ! ATN3 ! ATN2 ! ATN1 ! ATN0 !
2971                    ;                    READ ONLY                        !
2972                    ;-----------------------------------------------------
2973
2974                    ;  7      6      5      4      3      2      1      0
2975                    ;-----------------------------------------------------
2976                    ; UN  ! OF6  ! OF5  ! OF4  ! OF3  ! OF2  ! OF1  ! OF0  !
2977                    ;  ?  !              READ/WRITE                       !
2978                    ;-----------------------------------------------------
2979
2980                    ;RKWC (WORD COUNT)
2981                    ; 15     14     13     12     11     10     9      8
2982                    ;-----------------------------------------------------
2983                    ;WC15 ! WC14 ! WC13 ! WC12 ! WC11 ! WC10 ! WC9  ! WC8  !
2984                    ;                    READ/WRITE                       !
2985                    ;-----------------------------------------------------
2986
2987                    ;  7      6      5      4      3      2      1      0
2988                    ;-----------------------------------------------------
2989                    ; WC7 ! WC6  ! WC5  ! WC4  ! WC3  ! WC2  ! WC1  ! WC0  !
2990                    ;                    READ/WRITE                       !
2991                    ;-----------------------------------------------------
2992
2993                    ;RKBA (BUS ADDRESS)
2994                    ; 15     14     13     12     11     10     9      8
2995                    ;-----------------------------------------------------
2996                    ;BA15 ! BA14 ! BA13 ! BA12 ! BA11 ! BA10 ! BA9  ! BA8  !
2997                    ;                    READ/WRITE                       !
2998                    ;-----------------------------------------------------
2999
3000                    ;  7      6      5      4      3      2      1      0
3001                    ;-----------------------------------------------------
3002                    ; BA7 ! BA6  ! BA5  ! BA4  ! BA3  ! BA2  ! BA1  ! BA0  !
3003                    ;                    READ/WRITE                !ALWYS0!
3004                    ;-----------------------------------------------------
```

```
3005                                    ;RKER (ERROR REGISTER)
3006                                    ; 15     14     13     12     11     10     9      8
3007                                    ;-----------------------------------------------------
3008                                    ; DCK ! UNS  ! OPI  ! DTE  ! WLE  ! IDAE ! COE  ! HVRC !
3009                                    ;                    READ ONLY                        !
3010                                    ;-----------------------------------------------------
3011
3012                                    ;  7     6      5      4      3      2      1      0
3013                                    ;-----------------------------------------------------
3014                                    ; BSE ! ECH  ! DTYPE! FMTE ! DRPAR! ILF  ! SKI  ! ILC  !
3015                                    ;                    READ ONLY                        !
3016                                    ;-----------------------------------------------------
3017
3018                                    ;RKDS (DRIVE STATUS)
3019                                    ; 15     14     13     12     11     10     9      8
3020                                    ;-----------------------------------------------------
3021                                    ;SVAL ! DSC  ! PIP  ! SPON ! WRL  !  UN  !  UN  ! DTP  !
3022                                    ;                    READ ONLY                        !
3023                                    ;-----------------------------------------------------
3024
3025                                    ;  7     6      5      4      3      2      1      0
3026                                    ;-----------------------------------------------------
3027                                    ;DRDY !  VV  ! DROT ! SPLS ! ACLO ! OFSET!  UN  ! DRA  !
3028                                    ;                    READ ONLY                        !
3029                                    ;-----------------------------------------------------
3030
3031                                    ;RKMR1 (MAINTENANCE REG 1)
3032                                    ; 15    14     13     12     11     10     9      8
3033                                    ;-----------------------------------------------------
3034                                    ;RD    ! WRT  ! ECCW ! PCD  ! PCA  ! MEWD ! MERD ! MCLK !
3035                                    ;GATE  ! GATE !      !      !      !      !      !      !
3036                                    ;                 READ ONLY                 ! READ/WRITE !
3037                                    ;-----------------------------------------------------
3038
3039                                    ;  7     6      5      4      3      2      1      0
3040                                    ;-----------------------------------------------------
3041                                    ;MIND ! MSP  ! DMD  ! PAT  ! MS3  ! MS2  ! MS1  ! MS0  !
3042                                    ;                    READ/WRITE                       !
3043                                    ;-----------------------------------------------------
```

```
3044                                    ;RKECC/PAT
3045                                    ; 15     14     13     12     11     10     9      8
3046                                    ;-----------------------------------------------------
3047                                    ; UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
3048                                    ;                    READ ONLY                        !
3049                                    ;-----------------------------------------------------
3050
3051                                    ;  7      6      5      4      3      2      1      0
3052                                    ;-----------------------------------------------------
3053                                    ;EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
3054                                    ;                    READ ONLY                        !
3055                                    ;-----------------------------------------------------
3056
3057                                    ;RKECC/POS
3058                                    ; 15     14     13     12     11     10     9      8
3059                                    ;-----------------------------------------------------
3060                                    ; UN  !  UN  !  UN  ! EPO12! EPO11! EPO10! EPO9 ! EPO8 !
3061                                    ;                    READ ONLY                        !
3062                                    ;-----------------------------------------------------
3063
3064                                    ;  7      6      5      4      3      2      1      0
3065                                    ;-----------------------------------------------------
3066                                    ;EPO7 ! EPO6 ! EPO5 ! EPO4 ! EPO3 ! EPO2 ! EPO1 ! EPO0 !
3067                                    ;                    READ ONLY                        !
3068                                    ;-----------------------------------------------------
3069
3070                                    ;DRIVE STATUS INFORMATION
3071
3072                                    ;LINE A MESSAGE 00
3073                                    ; 15     14     13     12     11     10     9      8
3074                                    ;-----------------------------------------------------
3075                                    ; PAR ! DSC  ! PIP  ! SPON ! WRLK ! OFFON! FMT  ! DRTP !
3076                                    ;-----------------------------------------------------
3077
3078                                    ;  7      6      5      4      3      2      1      0
3079                                    ;-----------------------------------------------------
3080                                    ;DRDY !  VV  ! DRA  !  NU  !  NU  ! DRIVE SELECT CODE  !
3081                                    ;-----------------------------------------------------
```

```
                        ;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  !MD 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
```

```
3176    000012          RKDS=   12              ;DRIVE STATUS REGISTER
3177    000014          RKER=   14              ;ERROR REGISTER
3178    000016          RKASOF= 16              ;ATTENTION SUMMARY AND OFFSET REGISTER
3179    000020          RKDC=   20              ;DESIRED CYLINDER REGISTER
3180    000020          RKDCYL= 20              ;DESIRED CYLINDER REGISTER
3181    000024          RKDB=   24              ;DATA BUFFER
3182    000026          RKMR1=  26              ;MAINTENANCE REGISTER 1
3183    000034          RKMR2=  34              ;MAINTENANCE REGISTER 2
3184    000036          RKMR3=  36              ;MAINTENANCE REGISTER 3
3185    000030          RKPOS=  30              ;ECC POSITION INFORMATION
3186    000030          RKECPS= 30              ;ECC POSITION INFORMATION
3187    000032          RKPAT=  32              ;ECC PATTERN INFORMATION
3188    000032          RKECPT= 32              ;ECC PATTERN INFORMATION
3189
3190                    .SBTTL  DRIVE COMMANDS
3191
3192    000101          SELDRV= 101             ;SELECT DRIVE
3193    000103          PACK=   103             ;PACK ACKNOWLEDGE
3194    000105          CLEAR=  105             ;DRIVE CLEAR
3195    000107          UNLOAD= 107             ;UNLOAD
3196    000111          SRTSPL= 111             ;START SPINDLE
3197    000113          RECAL=  113             ;RECALIBRATE
3198    000115          OFFSET= 115             ;OFFSET
3199    000117          SEEK=   117             ;SEEK
3200    000121          RDDATA= 121             ;READ DATA
3201    000123          WRDATA= 123             ;WRITE DATA
3202    000125          RDHEAD= 125             ;READ HEADER
3203    000127          WRHEAD= 127             ;WRITE HEADER AND DATA
3204    000131          WRTCHK= 131             ;WRITE CHECK
3205
3206                    ;       THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
3207                    ;       TO SIMULATE A SPECIFIC DESIRED OPERATION
3208
3209    000140          RELEAS= 140             ;RELEASE DRIVE
3210    000141          RDSTAT= 141             ;GET ALL STATUS FROM DRIVE
3211    000164          RDALHD= 164             ;READ ALL HEADERS
3212    000176          CONCLR= 176             ;CONTROLLER CLEAR (BIT 15 OF CS1)
3213    000177          SUBCLR= 177             ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
3214    000300          INTR=   300             ;GENERATE INTERRUPT TO CPU
3215
3216                    ;       DRIVER ISSUED SERVICE COMMANDS
3217
3218    000001          DR.SEL= 001             ;DRIVE SELECT
3219    000005          DR.CLR= 005             ;DRIVE CLEAR
3220
3221                    .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
3222
3223    000001          GO=     BIT0            ;GO BIT
3224    000100          IE=     BIT6            ;INTERRUPT ENABLE
3225    000200          RDY=    BIT7            ;CONTROLLER READY
3226    000400          BA16=   BIT8            ;BUS ADDRESS BIT 16
3227    001000          BA17=   BIT9            ;BUS ADDRESS BIT 17
3228    002000          CDT=    BIT10           ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
3229    004000          CTO=    BIT11           ;CONTROLLER TIMED OUT WAITING FOR
3230                                            ;  DRIVE RESPONSE
3231    010000          CFMT=   BIT12           ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
```

```
3232            020000                  SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
3233            040000                  DI=     BIT14                   ;DRIVE INTERRUPT
3234            100000                  CERR=   BIT15                   ;CONTROLLER ERROR
3235            100000                  CCLR=   BIT15                   ;CONTROLLER CLEAR
3236
3237                                    ;       THESE BIT DEFINITIONS ARE USED FOR ADDRESS
3238                                    ;       THE HIGH BYTE OF RKCS1
3239
3240            000001                  B.BA16= BIT0                    ;BUS ADDRESS BIT 16
3241            000002                  B.BA17= BIT1                    ;BUS ADDRESS BIT 17
3242            000004                  B.CDT=  BIT2                    ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
3243            000020                  B.CFMT= BIT4                    ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
3244
3245                                    .SBTTL  CONTROL AND STATUS REGISTER 2 BITS
3246
3247            000007                  DRVMSK= 7                       ;MASK FOR DRIVE SELECTION CODE
3248            000010                  DESL=   BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3249            000010                  RLS=    BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3250            000020                  BAI=    BIT4                    ;BUS ADDRESS INCREMENT INHIBIT
3251            000040                  CLR=    BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
3252            000040                  SCLR=   BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
3253            000100                  IR=     BIT6                    ;INPUT READY
3254            000200                  OR=     BIT7                    ;OUTPUT READY
3255            000400                  UFE=    BIT8                    ;UNIT FIELD ERROR
3256            001000                  MDS=    BIT9                    ;MULTIPLE DRIVE SELECT
3257            002000                  PGE=    BIT10                   ;PROGRAMMING ERROR
3258            004000                  NEM=    BIT11                   ;NON-EXISTENT MEMORY
3259            010000                  NED=    BIT12                   ;NON-EXISTENT DRIVE
3260            020000                  UPE=    BIT13                   ;UNIBUS PARITY ERROR
3261            040000                  WCE=    BIT14                   ;WRITE CHECK ERROR
3262            100000                  DLT=    BIT15                   ;DATA LATE ERROR
3263
3264                                    .SBTTL  ERROR REGISTER BIT DEFINITION
3265
3266            000001                  ILC=    BIT0                    ;ILLEGAL FUNCTION CODE
3267                                    ;*ILF=  BIT0                    ;ILLEGAL FUNCTION CODE
3268            000002                  SKI=    BIT1                    ;SEEK INCOMPLETE
3269            000004                  ILF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
3270            000004                  NXF=    BIT2                    ;ILLEGAL DRIVE FUNCTION
3271            000010                  DRPAR=  BIT3                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
3272            000020                  FMTE=   BIT4                    ;FORMAT ERROR
3273            000040                  DTYE=   BIT5                    ;DRIVE TYPE ERROR
3274            000100                  ECH=    BIT6                    ;ECC HARD
3275            000200                  BSE=    BIT7                    ;BAD SECTOR ERROR
3276            000400                  HCRC=   BIT8                    ;HEADER CRC ERRROR
3277            000400                  HVRC=   BIT8                    ;HEADER VRC ERRROR
3278            001000                  COE=    BIT9                    ;CYLINDER ADDRESS OVERFLOW ERROR
3279            002000                  IDAE=   BIT10                   ;INVALID DISK ADDRESS ERROR
3280            004000                  WLE=    BIT11                   ;WRITE LOCK ERROR
3281            010000                  DTE=    BIT12                   ;DRIVE TIMING ERROR
3282            020000                  OPI=    BIT13                   ;OPERATION (SEARCH) INCOMPLETE
3283            040000                  UNS=    BIT14                   ;DRIVE UNSAFE
3284            100000                  DCK=    BIT15                   ;DATA CHECK
3285
3286                                    .SBTTL  STATUS REGISTER BIT DEFINITION
3287
```

```
                        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                  DRDY=   BIT7            ;DRIVE READY
000400                  DDT=    BIT8            ;DRIVE TYPE (0=RK06,1=RK07)
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
```

```
3344            010000                  S.DCLO= BIT12                   ;DC LOW
3345            020000                  S.DROT= BIT13                   ;DRIVE OFF TRACK
3346            040000                  S.UNS=  BIT14                   ;DRIVE UNSAFE
3347
3348                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3349
3350            000020                  S.XDOK= BIT4                    ;TRANSDUCER OK
3351            000040                  S.HDHM= BIT5                    ;HEADS HOME
3352            000100                  S.BRHM= BIT6                    ;BRUSHES HOME
3353            000200                  S.DOOR= BIT7                    ;DOOR INTERLOCKED
3354            000400                  S.CART= BIT8                    ;CARTRAGE INTERLOCK
3355            001000                  S.SPOK= BIT9                    ;SPEED OK
3356            002000                  S.FWD=  BIT10                   ;FORWARD
3357            004000                  S.REV=  BIT11                   ;REVERSE
3358            010000                  S.LOAD= BIT12                   ;HEADS LOADING
3359            020000                  S.RTZ=  BIT13                   ;RETURN TO ZERO
3360            040000                  S.UNLD= BIT14                   ;HEADS UNLOADING
3361
3362                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3363
3364            000020                  S.SECT= BIT4                    ;SECTOR ERROR
3365            000040                  S.WCLK= BIT5                    ;WRITE CLOCK AND NO WRITE GATE
3366            000100                  S.WGAT= BIT6                    ;WRITE GATE AND NO TRANSISTIONS
3367            000200                  S.HDFL= BIT7                    ;HEAD FAULT
3368            000400                  S.MHD=  BIT8                    ;MULTIPLE HEAD SELECT
3369            001000                  S.XERR= BIT9                    ;INDEX ERROR
3370            002000                  S.DIB=  BIT10                   ;DIBIT ERROR
3371            004000                  S.PLO=  BIT11                   ;PLO ERROR
3372            010000                  S.NMOV= BIT12                   ;SEEK AND NO MOTION
3373            020000                  S.LIMD= BIT13                   ;LIMIT DETECT ON SEEK
3374            040000                  S.BRKE= BIT14                   ;SERVO-BRAKE
3375
3376                                    .SBTTL  COMMON MASKS
3377
3378            000007                  M.DRV=  7                       ;DRIVE CODE
3379            100000                  M.PAR=  BIT15                   ;PARITY
3380            000003                  M.ID=   3                       ;BYTE ID
3381            017760                  M.CDIF= 17760                   ;CYLINDER DIFFERENCE/OFFSET
3382            017760                  M.CADD= 17760                   ;CYLINDER ADDRESS
3383            077770                  M.SER=  77770                   ;DRIVE SERIAL NUMBER
3384            000760                  M.SECT= 760                     ;SECTOR COUNT
3385            007000                  M.HEAD= 7000                    ;HEAD DECODE
```

```
                        .SBTTL  PARAMETER BLOCK ALLOCATION

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

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

                        ;       THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
                        ;         TO THE RK06/RK07 DRIVER

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

                        .SBTTL  PROGRAM DEVICE STATUS REGISTER DEFINITION

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

```
3442            000100                  CMDTO=  BIT6                    ;NO TERMINATION TO COMMAND FOR AT
3443                                                                    ;  LEAST 1 SECOND
3444            000200                  W.WCK=  BIT7                    ;WRITE FOR WRITE WRITE CHECK
3445            000400                  NOCHK=  BIT8                    ;NO CHECK, DO NOT SET INTERRUPT ENABLE
3446            001000                  PBSVAL= BIT9                    ;PARAMETER STATUS WORDS VALID
3447                                                                    ;  (SET WHEN ERROR TERMINATION OR
3448                                                                    ;  READ STATUS COMMAND)
3449            002000                  DRPDRV= BIT10                   ;DROP DRIVE FROM TEST SEQUENCE
3450            004000                  NODSC=  BIT11                   ;ATTENTION SET BUT DCS AND FAULT RESET
3451            010000                  DRVSZD= BIT12                   ;DRIVE SEIZED BY OTHER PORT
3452            020000                  E.UNLD= BIT13                   ;DRIVE UNLOADED DUE TO ERROR
3453            040000                  C.INIT= BIT14                   ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
3454            100000                  DTBAII= BIT15                   ;INHIBIT BUS ADDRESS INCREMENT
3455
3456                                    .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
3457
3458                                    ;       THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
3459                                    ;         FROM THE DRIVER TO THE CALLING PROGRAM
3460
3461            000016                  P.CS1=  16                      ;COMMAND AND STATUS REGISTER 1
3462            000020                  P.CS2=  20                      ;COMMAND AND STATUS REGISTER 2
3463            000022                  P.WCR=  22                      ;WORD COUNT REGISTER
3464            000024                  P.BAR=  24                      ;BUS ADDRESS REGISTER
3465            000026                  P.DTS=  26                      ;DESIRED TRACK SECTOR REGISTER
3466            000030                  P.DCYL= 30                      ;DESIRED CYLINDER REGISTER
3467            000032                  P.ASOF= 32                      ;ATTENTION SUMMARY/OFFSET REGISTER
3468            000034                  P.ER=   34                      ;ERROR REGISTER
3469            000036                  P.DS=   36                      ;STATUS REGISTER
3470            000040                  P.A00=  40                      ;MESSAGE A STATUS BYTE 00
3471            000042                  P.B00=  42                      ;MESSAGE B STATUS BYTE 00
3472            000044                  P.A01=  44                      ;MESSAGE A STATUS BYTE 01
3473            000046                  P.B01=  46                      ;MESSAGE B STATUS BYTE 01
3474            000050                  P.A10=  50                      ;MESSGGE A STATUS BYTE 10
3475            000052                  P.B10=  52                      ;MESSAGE B STATUS BYTE 10
3476            000054                  P.A11=  54                      ;MESSAGE A STATUS BYTE 11
3477            000056                  P.B11=  56                      ;MESSAGE B STATUS BYTE 11
3478            000060                  P.EPOS= 60                      ;ECC POSITION INFORMATION
3479            000062                  P.EPAT= 62                      ;ECC PATTERN INFORMATION
3480
3481                                    .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
3482
3483    002630      000                 PARM0:  .BYTE   0               ;DRIVE NUMBER
3484    002631      000                         .BYTE   0               ;COMMAND
3485    002632   000000                         .WORD   0               ;CYLINDER ADDRESS
3486    002634      000                         .BYTE   0               ;SECTOR ADDRESS
3487    002635      000                         .BYTE   0               ;TRACK ADDRESS
3488    002636      000                         .BYTE   0               ;OFFSET VALUE
3489    002637      000                         .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3490    002640   000000                         .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3491    002642   000000                         .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3492    002644   000000                         .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
3493    002646   000000                         .WORD   0               ;COMMAND AND STATUS REGISTER 1
3494    002650   000000                         .WORD   0               ;COMMAND AND STATUS REGISTER 2
3495    002652   000000                         .WORD   0               ;WORD COUNT REGISTER
3496    002654   000000                         .WORD   0               ;BUS ADDRESS REGISTER
3497    002656   000000                         .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
```

```
3498  002660  000000                    .WORD   0               ;DESIRED CYLINDER REGISTER
3499  002662  000000                    .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3500  002664  000000                    .WORD   0               ;ERROR REGISTER
3501  002666  000000                    .WORD   0               ;STATUS REGISTER
3502  002670  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3503  002672  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3504  002 74  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3505  002676  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3506  002700  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3507  002702  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3508  002704  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3509  002706  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3510  002710  000000                    .WORD   0               ;ECC POSITION INFORMATION
3511  002712  000000                    .WORD   0               ;ECC PATTERN INFORMATION
3512
3513                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
3514
3515  002714     000          PARM1:  .BYTE   0               ;DRIVE NUMBER
3516  002715     000                  .BYTE   0               ;COMMAND
3517  002716  000000                  .WORD   0               ;CYLINDER ADDRESS
3518  002720     000                  .BYTE   0               ;SECTOR ADDRESS
3519  002721     000                  .BYTE   0               ;TRACK ADDRESS
3520  002722     000                  .BYTE   0               ;OFFSET VALUE
3521  002723     000                  .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3522  002724  000000                  .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3523  002726  000000                  .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3524  002730  000000                  .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
3525  002732  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 1
3526  002734  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 2
3527  002736  000000                  .WORD   0               ;WORD COUNT REGISTER
3528  002740  000000                  .WORD   0               ;BUS ADDRESS REGISTER
3529  002742  000000                  .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
3530  002744  000000                  .WORD   0               ;DESIRED CYLINDER REGISTER
3531  002746  000000                  .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3532  002750  000000                  .WORD   0               ;ERROR REGISTER
3533  002752  000000                  .WORD   0               ;STATUS REGISTER
3534  002754  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3535  002756  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3536  002760  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3537  002762  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3538  002764  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3539  002766  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3540  002770  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3541  002772  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3542  002774  000000                  .WORD   0               ;ECC POSITION INFORMATION
3543  002776  000000                  .WORD   0               ;ECC PATTERN INFORMATION
3544
3545                        .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
3546
3547  003000  000000        T.CS1:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3548                                                        ;  REGISTER 1
3549  003002  000000        T.CS2:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3550                                                        ;  REGISTER 2
3551  003004  000000        T.WCR:  .WORD   0               ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
3552  003006  000000        T.BA:   .WORD   0               ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
3553  003010  000000        T.DA:   .WORD   0               ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
3554  003012  000000                T.DC:   .WORD   0               ;TEMPORARY STORAGE FOR DRIVE CYLINDER
3555  003014  000000                T.ASOF: .WORD   0               ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
3556                                                                ;  AND OFFSET
3557  003016  000000                T.ER:   .WORD   0               ;TEMPORARY STORAGE FOR ERROR REGISTER
3558  003020  000000                T.DS:   .WORD   0               ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3559  003022  000000                T.MR1:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3560  003024  000000                T.MR2:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3561  003026  000000                T.MR3:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3562  003030  000000                T.POS:  .WORD   0               ;TEMPORARY STORAGE FOR ECC POSITION
3563  003032  000000                T.PAT:  .WORD   0               ;TEMPORARY STORAGE FOR ECC PATTERN
3564  003034  000000                T.DB:   .WORD   0               ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
3565
3566                                .SBTTL  DRIVER PARAMERTERS
3567
3568  003036  177440                RKBAS:  .WORD   177440          ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3569  003040  000210                RKVEC:  .WORD   210             ;ADDRESS OF R611 VECTOR
3570  003042  000240                RKPRI:  .WORD   PR5             ;RK611 INTERRUPT PRIORITY
3571  003044  042700                A.NORM: ERRFRE                  ;ADDRESS OF NORMAL RETURN FROM DRIVER
3572  003046  044156                A.ABNL: ERRHDL                  ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
3573  003050  043442                A.CONT: CONERR                  ;ADDRESS OF CONTROLLER ERROR RETURN
3574  003052  000000                E.CONT: .WORD   0               ;CONTROLLER ERROR STATUS
3575                                                                ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
3576                                                                ;  IS INITIATED.  IF A CONTROLLER ERROR
3577                                                                ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
3578                                                                ;  USED:
3579                                                                ;
3580          000001                E.CCLR= BIT0                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
3581          000002                E.NOAT= BIT1                    ;NO ATTENTION IN ATTENTION SUMMARY REG
3582          000004                E.UATT= BIT2                    ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3583          000010                E.UDAT= BIT3                    ;UNEXPECTED DATA TYPE ERROR
3584          000020                E.CLAT= BIT4                    ;ATTENTION DID NOT RESET WITH CLEAR
3585          000040                E.SCLR= BIT5                    ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
3586                                                                ;  ATTENTION
3587          000100                E.ILLD= BIT6                    ;ILLEGAL DRIVER COMMAND
3588          000400                E.DLT=  BIT8                    ;DATA LATE WHEN UNLOADING HEADER
3589          001000                E.CERR= BIT9                    ;CONTROLLER ERROR DURING DRIVER SERVICING
3590          002000                E.DPAR= BIT10                   ;DRIVE DETECTED PARITY ERROR
3591          040000                E.CMTO= BIT14                   ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3592          100000                E.MDS=  BIT15                   ;MULTIPLE DRIVE SELECT
3593
3594  003054  000000                O.WAIT: .WORD   0               ;PARAMETER BLOCK OF THE DRIVE
3595                                                                ;WAITING FOR COMMAND COMPLETION
3596  003056  000400                W.MTIM: .WORD   400             ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
3597  003060  000400                W.MILI: .WORD   400             ;16 MILISECOND TIME FOR PROGRAM
3598                                                                ;
3599                                                                ;       CPU     VALUE
3600                                                                ;       ---     -----
3601                                                                ;
3602                                                                ;       11/05   100
3603                                                                ;       11/10
3604                                                                ;       11/20
3605                                                                ;       11/34
3606                                                                ;       11/40
3607                                                                ;       11/45   400
3608                                                                ;       11/50
3609                                                                ;       11/70
```

```
3610  003062  000300                  W.SEC:  .WORD   300             ;SECOND COUNT COUNT FOR ALL COMMANDS
3611                                                                  ;  EXCEPT START SPINDLE
3612  003064  003000                  W.8SEC: .WORD   3000            ;8 SECOND FOR DRIVE CYCLE DOWN
3613  003066  030000                  W.MIN:  .WORD   30000           ;MINUTE TIME FOR START SPINDLE
3614  003070  000000                  HDR.AD: .WORD   0               ;ADDRESS USED FOR READ ALL HEADERS
3615  003072  000000                  HDR.CT: .WORD   0               ;NUMBER OF HEADERS LEFT TO READ FOR READ
3616                                                                  ;  ALL HEADERS
3617  003074     000                  I.ISRL: .BYTE   0               ;INTERRUPT OR RELEASED COMMAND ISSUED
3618  003075     002     004     010  H.HEAD: .BYTE   2,4,10          ;HEAD DECODES
3619  003100     000                  W.TIME: .BYTE   0               ;DRIVES BEING WATCH-DOG TIMED
3620
3621                                  .SBTTL  INTERRUPT MASKS
3622
3623  003101     000                  INTMSK: .BYTE   0               ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
3624
3625                                  ;       INTERRUPT MASK TABLE
3626
3627  003102     001                  I.DRV:  .BYTE   1               ;INTERRUPT MASK FOR DRIVE 0
3628  003103     002                          .BYTE   2               ;INTERRUPT MASK FOR DRIVE 1
3629  003104     004                          .BYTE   4               ;INTERRUPT MASK FOR DRIVE 2
3630  003105     010                          .BYTE   10              ;INTERRUPT MASK FOR DRIVE 3
3631  003106     020                          .BYTE   20              ;INTERRUPT MASK FOR DRIVE 4
3632  003107     040                          .BYTE   40              ;INTERRUPT MASK FOR DRIVE 5
3633  003110     100                          .BYTE   100             ;INTERRUPT MASK FOR DRIVE 6
3634  003111     200                          .BYTE   200             ;INTERRUPT MASK FOR DRIVE 7
3635
3636                                  .SBTTL  PARAMETER BLOCK TABLE
3637
3638  003112  002630                  PBLKT:  PARM0                   ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
3639                                                                  ;DRIVE CALL.  MUST BE LOADED INTO PBLKT
3640
3641
3642                                  .SBTTL  TIME FOR WATCH-DOG TIMER
3643
3644  003114  000000                  W.DRV:  .WORD   0               ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
3645                                  .SBTTL  PROGRAM SPECIFIC RESERVED LOCATIONS
3646  003116     000                  MDFLAG: .BYTE   0               ;FLAG TO INDIC. DEFLT OR PARAM MODE
3647  003117     000                  XXDPCH: .BYTE   0               ;XXDP CHAIN MODE FLAG
3648  003120     000                  TSTING: .BYTE   0               ;CURRENTLY RUNNING TESTS IF = 1
3649  003121     000                  DERCNT: .BYTE   0               ;DATA ERROR COUNT
3650  003122     000                  OPCOMP: .BYTE   0               ;OPERATION COMPLETE FLAG
3651  003123     000                  DONE:   .BYTE   0               ;DONE SWITCH
3652  003124     000                  TYPFMT: .BYTE   0               ;DRIVE TYPE & FORMAT CONTROL
3653  003125     000                  FORMAT: .BYTE   0               ;DRIVE FORMAT IN BIT 4 OF BYTE
3654  003126     000                  ERRCNT: .BYTE   0               ;ERROR COUNT
3655  003127     004                  ERRLMT: .BYTE   4               ;ERROR LIMIT
3656  003130     000                  DRVERS: .BYTE   0               ;ERROR COUNT FOR CURRENT DRIVE
3657  003131     000                  OPCONT: .BYTE   0               ;OPERATION CONTROL SWITCHES
3658  003132     000                  PCLKF:  .BYTE   0               ;IF BYTE=1, KW11-P CLOCK IS PRESENT
3659  003133     000                  XOVLAD: .BYTE   0               ;FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
3660  003134     000                  XDPSVD: .BYTE   0               ;FLAG = 1 IF XXDP IS SAVED
3661  003135     000                  DULACS: .BYTE   0               ;FLAG=1 IF DUAL ACCESS TEST
3662  003136     000                  DRNAFG: .BYTE   0               ;=1 INDICATES DRIVE SIEZED BY OTHER PORT
3663  003137     000                  REISSU: .BYTE   0               ;DUAL-ACC FLAG TO RE-ISSUE COMMAND
3664  003140     000                  WCEFLG: .BYTE   0               ;WRITE CHECK ERROR FLAG
3665  003141     000                  DLTFLG: .BYTE   0               ;DATA LATE ERROR FLAG
```

```
  3666  003142  000                 NORTRY: .BYTE   0               ;"NO-RETRY" FLAG
  3667  003143  000                 UBMPRS: .BYTE   0               ;UNIBUS MAP PRESENT IF = 1
  3668  003144  000                 MEMABT: .BYTE   0               ;SET BYTE = 1 FOR NO ABORT
  3669                                                              ;    ON MEMORY PARITY ERRORS
  3670  003145  000                 HLPOVL: .BYTE   0               ;HELP FILE OVERLAID INDICATOR
  3671  003146  000                 NOTYPE: .BYTE   0               ;INDICATES PROGRAM JUST LOADED IF 0
  3672          000001              WHDSW=BIT0                      ;WRITE HEADER & DATA SWITCH
  3673          000002              VFHDSW=BIT1                     ;VERIFY HEADERS SWITCH
  3674          000004              WCDASW=BIT2                     ;WRITE CHECK DATA SWITCH
  3675          000010              RCDASW=BIT3                     ;READ CHECK DATA SWITCH
  3676          000020              OREQSW=BIT4                     ;OFFSET REQUIRED SWITCH
  3677          003150                      .EVEN
  3678  003150  000000              AUTOFG  .WORD   0       ;SPECIAL FLAG
  3679
  3680  003152  177546              CKS     .WORD   177546          ;KW11-L CLOCK STATUS REGISTER
  3681  003154  172540              PKS     .WORD   172540          ;KW11-P CONTROL AND STATUS REGISTER
  3682  003156  172542              PKSB    .WORD   172542          ;KW11-P COUNT SET BUFFER REGISTER
  3683  003160  172544              PKAB    .WORD   172544          ;KW11-P COUNTER REGISTER
  3684  003162  000100              LCVEC   .WORD   100             ;KW11-L VECTOR STORAGE
  3685  003164  000104              PCVEC:  .WORD   104             ;KW11-P VECTOR STORAGE
  3686          000114              MEMVEC=114                      ;MEMORY PARITY TRAP VECTOR
```

```
3687  003166  000000                  TCONLO: .WORD   0               ;LO BITS OF CALIBRATION TIME CONSTANT
3688  003170  000000                  TCONHI: .WORD   0               ;HI BITS OF CALIB TIME CONST
3689  003172  000000                  SUMLO1: .WORD   0               ;LO BITS OF FORWARD TIME SUM
3690  003174  000000                  SUMHI1: .WORD   0               ;HI BITS OF FORWARD TIME SUM
3691  003176  000000                  SAVPAR: .WORD   0               ;SAVE WORD FOR PAR CONSTANT
3692  003200  000000                  SAVWRD: .WORD   0               ;SCRATCH WORD
3693  003202  000000                  CYL:    .WORD   0               ;STARTING CYL FOR MULTI-DRV TEST
3694  003204  000010                  CYLLST: .BLKW   ^D8             ;LIST OF PSEUDO-RAND CYL ADDRESSES
3695  003224  000400                  BSSOFT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM SOFTWARE
3696  004224  000400                  BSFACT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM FACTORY
3697  005224  000000  000000  000000  PRVCMD: .WORD   0,0,0,0,0,0     ;PREVIOUS COMMAND STORAGE
3698  005232  000000  000000  000000
3699  005240  000000  000000  000000  COMSTR: .WORD   0,0,0,0,0,0     ;CURRENT COMMAND STORAGE
3700  005246  000000  000000  000000
3701  005254  000000                  PRMPL0: .WORD   0               ;PREV. U.B. MAP REG 0
3702  005256  000000                  PRMPH0: .WORD   0
3703  005260  000000                  CRMPL0: .WORD   0               ;CURRENT U.B. MAP REG 0
3704  005262  000000                  CRMPH0: .WORD   0
3705  005264  000102                  BUFF0:  .BLKW   ^D66            ;OUTPUT BUFFER 1
3706  005470  000000                  LOWOCT: .WORD   0               ;LOW 16 BITS OF CONVERTED BINARY NUMBER
3707  005472  000000                  HIGOCT: .WORD   0               ;HIGH BITS OF CONVERTED BINARY NO.
3708  005474  000000                  RECODE: .WORD   0               ;RECOVERY CODE WORD
3709  005476  000000                  ERRCOM: .WORD   0               ;ERROR COMMAND
3710  005500  000000                  DRIVE:  .WORD   0               ;NO.  OF  DRIVE IN USE
3711  005502  000000                  STALLS: .WORD   0               ;CURRENT NO. OF UNIT STALLS TO APPLY
3712  005504  000000                  CYLNDR: .WORD   0               ;CURRENT CYLINDER NUMBER
3713  005506  000000                  FS:     .WORD   0               ;FIRST SECTOR LIMIT
3714  005510  000000                  LS:     .WORD   0               ;LAST SECTOR LIMIT
3715  005512  000000                  NCYL1:  .WORD   0               ;NEXT CYL SCRATCH WORD
3716  005514  000000                  NCYL2:  .WORD   0               ;NEXT CYL SCRATCH WORD
3717  005516  000000                  OFINUS: .WORD   0               ;OFFSET IN USE
3718  005520  000000                  TRACK:  .WORD   0               ;TRACK IN USE
3719  005522  000000                  INTCHR: .WORD   0               ;TTY INTERRUPT INPUT WORD
3720  005524  000000                  SELECT: .WORD   0               ;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3721  005526  000000                  ONLINE: .WORD   0               ;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3722  005530  000000                  NEWON:  .WORD   0               ;NEW LOOK AT ONLINE DRIVES
3723  005532  000000                  HDALI:  .WORD   0               ;TAG USED FOR INTERNAL USE
3724  005534  000000                  SCRACH: .WORD   0               ;ALL-PURPOSE SCRATCH WORD
3725  005536  000000                  PATRN:  .WORD   0               ;DATA PATTERN LIST WORD
3726  005540  000000                  XXDPAD: .WORD   0               ;STARTING ADDRESS OF XXDP LOADER
3727  005542  000002                  XDPSAV: .BLKW   2               ;XXDP PHYSICAL SAVE ADDRESS
3728  005546  000000                  PLOFST: .WORD   0               ;(+) OFFSET VALUE
3729  005550  000000                  NGOFST: .WORD   0               ;(-) OFFSET VALUE
3730  005552  000005                  SAVPRS: .BLKW   5               ;SAVE INITIAL PARAMS FOR XFER
3731  005564  000000                  WDSXFR: .WORD   0               ;NO. OF WORDS ACTUALLY XFERRED
3732  005566  000000                  FINCYL: .WORD   0               ;FINAL CYLINDER ADRS
3733  005570     000                  FINTRK: .BYTE   0               ;FINAL TRACK ADRS
3734  005571     000                  FINSEC: .BYTE   0               ;FINAL SECTOR ADRS
3735  005572  000000                  LASTWC: .WORD   0               ;ACTUAL FINAL WC
3736  005574  157776                  MAHILM: .WORD   157776          ;MA UPPER LIMIT
3737  005576  000077                          .WORD   77
3738  005600  065636                  MA:     .WORD   RWBUF           ;LO BITS OF PHYS MEM ADRS
3739  005602  000000                          .WORD   0               ;HI BITS OF PHYS MEM ADRS
3740  005604  000002                  PMA:    .BLKW   2               ;PARTIAL TRANSFER MEMORY START ADRS
3741
3742
```

```
                                        ;LIST OF DESIRED DRIVES TO TEST (IF BYTE=0, DON'T TEST DRIVE)
005610                                  DRVLST:
005610  001                                     .BYTE   1               ;DRIVE 0
005611  001                                     .BYTE   1               ;DRIVE 1
005612  001                                     .BYTE   1               ;DRIVE 2
005613  001                                     .BYTE   1               ;DRIVE 3
005614  001                                     .BYTE   1               ;DRIVE 4
005615  001                                     .BYTE   1               ;DRIVE 5
005616  001                                     .BYTE   1               ;DRIVE 6
005617  001                                     .BYTE   1               ;DRIVE 7
                                        ;LIST OF DRIVE TYPES, 0=RK06, NON 0=RK07
005620                                  DTYLST:
005620  000                                     .BYTE   0               ;DRV 0
005621  000                                     .BYTE   0               ;DRV 1
005622  000                                     .BYTE   0               ;DRV 2
005623  000                                     .BYTE   0               ;DRV 3
005624  000                                     .BYTE   0               ;DRV 4
005625  000                                     .BYTE   0               ;DRV 5
005626  000                                     .BYTE   0               ;DRV 6
005627  000                                     .BYTE   0               ;DRV 7
```

```
                                        ;LIST OF ITERATION COUNTS FOR DESIRED TESTS (IF=0, TEST NOT RUN)
                                        ;MAXIMUM ALLOWABLE NUMBER OF ITERATIONS = 77776 (OCT)
005630                                  TSTLST:
005630  000002                                  .WORD   2               ;TEST 1
005632  000002                                  .WORD   2               ;TEST 2
005634  000002                                  .WORD   2               ;TEST 3
005636  000002                                  .WORD   2               ;TEST 4
005640  000002                                  .WORD   2               ;TEST 5
005642  000100                                  .WORD   100             ;TEST 6
```

```
                                        ;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
005644                                  DFLTST:
005644  000002                                  .WORD   2               ;TEST 1
005646  000002                                  .WORD   2               ;TEST 2
005650  000002                                  .WORD   2               ;TEST 3
005652  000002                                  .WORD   2               ;TEST 4
005654  000002                                  .WORD   2               ;TEST 5
005656  000100                                  .WORD   100             ;TEST 6
```

```
        000006  NMTSTS=<DFLTST-TSTLST>/2         ;TOTAL NO. OF AUTOMATIC TESTS
```

```
                                        ;OPERATING PARAMETER LIST
005660                                  PRMLST:
005660  000000  FC:     .WORD   0               ;FIRST CYLINDER
005662  000000  LC:     .WORD   0               ;LAST CYLINDER
005664  000000  FT:     .WORD   0               ;FIRST TRACK
005666  000002  LT:     .WORD   2               ;LAST TRACK
```

```
3799  005670  000000                  S0:     .WORD   0               ;FIRST SECTOR IF 20(DEC) SECTOR FMT
3800  005672  000023                  S1:     .WORD   23              ;LAST SECTOR IF 20(DEC) SECTOR FMT
3801  005674  000000                  S2:     .WORD   0               ;FIRST SECTOR IF 22(DEC) SECTOR FMT
3802  005676  000025                  S3:     .WORD   25              ;LAST SECTOR IF 22(DEC) SECTOR FMT
3803  005700  000000                  PT:     .WORD   0               ;DATA PATTERN SELECT WORD
3804  005702  000000                  CS:     .WORD   0               ;CONTROL SWITCH WORD
3805  005704  000000                  ST:     .WORD   0               ;NUMBER OF UNIT STALLS
3806
3807
3808                                  ;DEFAULT OPERATING PARAMETER VALUES
3809  005706                          PRDFLT:
3810  005706  000000                          .WORD   0               ;FC DEFAULT
3811  005710  000000                          .WORD   0               ;LC DEFAULT
3812  005712  000000                          .WORD   0               ;FT DEFAULT
3813  005714  000002                          .WORD   2               ;LT DEFAULT
3814  005716  000000                          .WORD   0               ;S0 DEFAULT
3815  005720  000023                          .WORD   23              ;S1 DEFAULT
3816  005722  000000                          .WORD   0               ;S2 DEFAULT
3817  005724  000025                          .WORD   25              ;S3 DEFAULT
3818  005726  000000                          .WORD   0               ;PT DEFAULT
3819  005730  000000                          .WORD   0               ;CS DEFAULT
3820  005732  000000                          .WORD   0               ;ST DEFAULT
3821
3822
3823
3824                                  ;OPERATING PARAMETER VALUE LOW AND HIGH LIMITS
3825  005734                          PRMLIM:
3826  005734  000000                          .WORD   0               ;FC LIMITS
3827  005736  000000                          .WORD   0
3828  005740  000000                          .WORD   0               ;LC LIMITS
3829  005742  000000                          .WORD   0
3830  005744  000000                          .WORD   0               ;FT LIMITS
3831  005746  000002                          .WORD   2
3832  005750  000000                          .WORD   0               ;LT LIMITS
3833  005752  000002                          .WORD   2
3834  005754  000000                          .WORD   0               ;S0 LIMITS
3835  005756  000023                          .WORD   23
3836  005760  000000                          .WORD   0               ;S1 LIMITS
3837  005762  000023                          .WORD   23
3838  005764  000000                          .WORD   0               ;S2 LIMITS
3839  005766  000025                          .WORD   25
3840  005770  000000                          .WORD   0               ;S3 LIMITS
3841  005772  000025                          .WORD   25
3842  005774  000000                          .WORD   0               ;PT LIMITS
3843  005776  177777                          .WORD   177777
3844  006000  000000                          .WORD   0               ;CS LIMITS
3845  006002  000062                          .WORD   000062
3846  006004  000000                          .WORD   0               ;ST LIMITS
3847  006006  177777                          .WORD   177777
3848
3849
3850
3851                                  ;ASCII PARAMETER MNEMONICS
3852  006010                          PRMNEM:
3853  006010  041506                          .ASCII  /FC/
3854  006012  041514                          .ASCII  /LC/
```

```
3855  006014  052106                  .ASCII  /FT/
3856  006016  052114                  .ASCII  /LT/
3857  006020  030123                  .ASCII  /S0/
3858  006022  030523                  .ASCII  /S1/
3859  006024  031123                  .ASCII  /S2/
3860  006026  031523                  .ASCII  /S3/
3861  006030  052120                  .ASCII  /PT/
3862  006032  051503                  .ASCII  /CS/
3863  006034  052123                  .ASCII  /ST/
3864
3865
3866
3867                            ;DATA PATTERN 00
3868                            ;       HI-LO FREQ. MIX
3869  006036                    PAT00:
3870  006036  177777                  177777
3871  006040  177777                  177777
3872  006042  177777                  177777
3873  006044  052525                  052525
3874  006046  052525                  052525
3875  006050  052525                  052525
3876  006052  177777                  177777
3877  006054  177777                  177777
3878  006056  052525                  052525
3879  006060  052525                  052525
3880  006062  177777                  177777
3881  006064  052525                  052525
3882  006066  177252                  177252
3883  006070  177252                  177252
3884  006072  172765                  172765
3885  006074  172765                  172765
3886
3887
3888
3889                            ;DATA PATTERN 01
3890                            ;       HI FREQ. PHASE MIX
3891  006076                    PAT01:
3892  006076  000000                  000000
3893  006100  000000                  000000
3894  006102  000000                  000000
3895  006104  177777                  177777
3896  006106  177777                  177777
3897  006110  177777                  177777
3898  006112  000000                  000000
3899  006114  000000                  000000
3900  006116  177777                  177777
3901  006120  177777                  177777
3902  006122  000000                  000000
3903  006124  177777                  177777
3904  006126  000000                  000000
3905  006130  177777                  177777
3906  006132  000000                  000000
3907  006134  177777                  177777
3908
3909
3910
```

```
3911                                    ;DATA PATTERN 02
3912                                    ;       LO FREQ. PHASE MIX
3913  006136                            PAT02:
3914  006136  052525                           052525
3915  006140  052525                           052525
3916  006142  052525                           052525
3917  006144  125252                           125252
3918  006146  125252                           125252
3919  006150  125252                           125252
3920  006152  052525                           052525
3921  006154  052525                           052525
3922  006156  125252                           125252
3923  006160  125252                           125252
3924  006162  052525                           052525
3925  006164  125252                           125252
3926  006166  052525                           052525
3927  006170  125252                           125252
3928  006172  052525                           052525
3929  006174  125252                           125252
3930
3931
3932
3933                                    ;DATA PATTERN 03
3934                                    ;       MAX. PRECOMP. PHASE MIX
3935  006176                            PAT03:
3936  006176  133333                           133333
3937  006200  066666                           066666
3938  006202  155555                           155555
3939  006204  155555                           155555
3940  006206  133333                           133333
3941  006210  066666                           066666
3942  006212  066666                           066666
3943  006214  155555                           155555
3944  006216  155555                           155555
3945  006220  133333                           133333
3946  006222  133333                           133333
3947  006224  133333                           133333
3948  006226  133333                           133333
3949  006230  133333                           133333
3950  006232  133333                           133333
3951  006234  133333                           133333
3952
3953
3954
3955                                    ;DATA PATTERN 04
3956                                    ;       ROTATING BOUNDARY PULSE PRECOMP.
3957  006236                            PAT04:
3958  006236  121105                           121105
3959  006240  150442                           150442
3960  006242  064221                           064221
3961  006244  132110                           132110
3962  006246  055044                           055044
3963  006250  026422                           026422
3964  006252  013211                           013211
3965  006254  105504                           105504
3966  006256  042642                           042642
```

```
                                                    021321
                                                    110550
                                                    044264
                                                    022132
                                                    011055
                                                    104426
                                                    042213


                        ;DATA PATTERN 05
                        ;       ROTATING CELL PULSE PRECOMP.
                        ;
006276                  PAT05:
006276  026455                  026455
006300  113226                  113226
006302  045513                  045513
006304  122645                  122645
006306  151322                  151322
006310  064551                  064551
006312  132264                  132264
006314  055132                  055132
006316  026455                  026455
006320  113226                  113226
006322  045513                  045513
006324  122645                  122645
006326  151322                  151322
006330  064551                  064551
006332  132264                  132264
006334  055132                  055132


                        ;DATA PATTERN 06
                        ;       ALL ZEROS
                        ;
006336                  PAT06:
006336  000000                  000000
006340  000000                  000000
006342  000000                  000000
006344  000000                  000000
006346  000000                  000000
006350  000000                  000000
006352  000000                  000000
006354  000000                  000000
006356  000000                  000000
006360  000000                  000000
006362  000000                  000000
006364  000000                  000000
006366  000000                  000000
006370  000000                  000000
006372  000000                  000000
006374  000000                  000000


                        ;DATA PATTERN 07
                        ;       ALL ONES
                        ;
006376                  PAT07:
```

```
 4023  006376  177777                    177777
 4024  006400  177777                    177777
 4025  006402  177777                    177777
 4026  006404  177777                    177777
 4027  006406  177777                    177777
 4028  006410  177777                    177777
 4029  006412  177777                    177777
 4030  006414  177777                    177777
 4031  006416  177777                    177777
 4032  006420  177777                    177777
 4033  006422  177777                    177777
 4034  006424  177777                    177777
 4035  006426  177777                    177777
 4036  006430  177777                    177777
 4037  006432  177777                    177777
 4038  006434  177777                    177777
 4039
 4040
 4041                              ;DATA PATTERN 08
 4042                              ;       SHIFTED 1 IN FIELD OF ZEROS
 4043  006436                      PAT08:
 4044  006436  000001                    000001
 4045  006440  000002                    000002
 4046  006442  000004                    000004
 4047  006444  000010                    000010
 4048  006446  000020                    000020
 4049  006450  000040                    000040
 4050  006452  000100                    000100
 4051  006454  000200                    000200
 4052  006456  000400                    000400
 4053  006460  001000                    001000
 4054  006462  002000                    002000
 4055  006464  004000                    004000
 4056  006466  010000                    010000
 4057  006470  020000                    020000
 4058  006472  040000                    040000
 4059  006474  100000                    100000
 4060
 4061
 4062
 4063                              ;DATA PATTERN 09
 4064                              ;       SHIFTED 0 IN FIELD OF ONES
 4065  006476                      PAT09:
 4066  006476  177776                    177776
 4067  006500  177775                    177775
 4068  006502  177773                    177773
 4069  006504  177767                    177767
 4070  006506  177757                    177757
 4071  006510  177737                    177737
 4072  006512  177677                    177677
 4073  006514  177577                    177577
 4074  006516  177377                    177377
 4075  006520  176777                    176777
 4076  006522  175777                    175777
 4077  006524  173777                    173777
 4078  006526  167777                    167777
```

```
 4079  006530  157777                    157777
 4080  006532  137777                    137777
 4081  006534  077777                    077777
 4082
 4083
 4084
 4085                              ;DATA PATTERN 10
 4086                              ;       ALTERNATING 0-1
 4087  006536                      ;PAT10:
 4088  006536  052525                    052525
 4089  006540  052525                    052525
 4090  006542  052525                    052525
 4091  006544  052525                    052525
 4092  006546  052525                    052525
 4093  006550  052525                    052525
 4094  006552  052525                    052525
 4095  006554  052525                    052525
 4096  006556  052525                    052525
 4097  006560  052525                    052525
 4098  006562  052525                    052525
 4099  006564  052525                    052525
 4100  006566  052525                    052525
 4101  006570  052525                    052525
 4102  006572  052525                    052525
 4103  006574  052525                    052525
 4104
 4105
 4106
 4107                              ;DATA PATTERN 11
 4108                              ;       ALTERNATING 1-0
 4109  006576                      ;PAT11:
 4110  006576  125252                    125252
 4111  006600  125252                    125252
 4112  006602  125252                    125252
 4113  006604  125252                    125252
 4114  006606  125252                    125252
 4115  006610  125252                    125252
 4116  006612  125252                    125252
 4117  006614  125252                    125252
 4118  006616  125252                    125252
 4119  006620  125252                    125252
 4120  006622  125252                    125252
 4121  006624  125252                    125252
 4122  006626  125252                    125252
 4123  006630  125252                    125252
 4124  006632  125252                    125252
 4125  006634  125252                    125252
 4126
 4127
 4128
 4129                              ;DATA PATTERN 12
 4130                              ;       SHIFTING ZEROS AND ONES
 4131  006636                      ;PAT12:
 4132  006636  000001                    000001
 4133  006640  000003                    000003
 4134  006642  000007                    000007
```

```
 4135  006644  000017                    000017
 4136  006646  000037                    000037
 4137  006650  000077                    000077
 4138  006652  000177                    000177
 4139  006654  000377                    000377
 4140  006656  000777                    000777
 4141  006660  001777                    001777
 4142  006662  003777                    003777
 4143  006664  007777                    007777
 4144  006666  017777                    017777
 4145  006670  037777                    037777
 4146  006672  077777                    077777
 4147  006674  177777                    177777
 4148
 4149
 4150
 4151                              ;DATA PATTERN 13
 4152                              ;       COMPOSITE ROTATING
 4153  006676                      PAT13:
 4154  006676  072307                    072307
 4155  006700  135143                    135143
 4156  006702  156461                    156461
 4157  006704  167230                    167230
 4158  006706  073514                    073514
 4159  006710  035646                    035646
 4160  006712  016723                    016723
 4161  006714  107351                    107351
 4162  006716  143564                    143564
 4163  006720  061672                    061672
 4164  006722  030735                    030735
 4165  006724  114356                    114356
 4166  006726  046167                    046167
 4167  006730  123073                    123073
 4168  006732  151453                    151453
 4169  006734  164616                    164616
 4170
 4171
 4172
 4173                              ;DATA PATTERN 14
 4174                              ;       PSEUDO-RANDOM (COMPUTED BY PROGRAM)
 4175  006736                      PAT14:
 4176  006736  000000                    .WORD
 4177  006740  000000                    .WORD
 4178  006742  000000                    .WORD
 4179  006744  000000                    .WORD
 4180  006746  000000                    .WORD
 4181  006750  000000                    .WORD
 4182  006752  000000                    .WORD
 4183  006754  000000                    .WORD
 4184  006756  000000                    .WORD
 4185  006760  000000                    .WORD
 4186  006762  000000                    .WORD
 4187  006764  000000                    .WORD
 4188  006766  000000                    .WORD
 4189  006770  000000                    .WORD
 4190  006772  000000                    .WORD
```

```
 4191  006774  000000                    .WORD
 4192
 4193
 4194                                    ;USER-DEFINED DATA PATTERN 15
 4195  006776                            PAT15:
 4196  006776  072307                          072307                  ;WORD 00
 4197  007000  135143                          135143                  ;WORD 01
 4198  007002  156461                          156461                  ;WORD 02
 4199  007004  167230                          167230                  ;WORD 03
 4200  007006  073514                          073514                  ;WORD 04
 4201  007010  035646                          035646                  ;WORD 05
 4202  007012  016723                          016723                  ;WORD 06
 4203  007014  107351                          107351                  ;WORD 07
 4204  007016  143564                          143564                  ;WORD 10
 4205  007020  061672                          061672                  ;WORD 11
 4206  007022  030735                          030735                  ;WORD 12
 4207  007024  114356                          114356                  ;WORD 13
 4208  007026  046167                          046167                  ;WORD 14
 4209  007030  123073                          123073                  ;WORD 15
 4210  007032  151453                          151453                  ;WORD 16
 4211  007034  164616                          164616                  ;WORD 17
 4212
 4213
 4214
 4215          057467                    MINLO1=^D24375                ;LO BITS OF SPEC'D MIN ROT. LATENCY
 4216          000000                    MINHI1=0                      ;HI BITS OF SPEC'D MIN ROT. LATENCY
 4217          062031                    MAXLO1=^D25625                ;LO BITS OF SPEC'D MAX ROT. LATENCY
 4218          000000                    MAXHI1=0                      ;HI BITS OF SPEC'D MAX ROT. LATENCY
 4219          017500                    MAXLO2=^D8000                 ;LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
 4220          000000                    MAXHI2=0                      ;HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
 4221          112160                    MAXLO3=^D38000                ;LO BITS OF SPEC'D MAX AVG SEEK TIME
 4222          000000                    MAXHI3=0                      ;HI BITS OF SPEC'D MAX AVG SEEK TIME
 4223          022370                    MAXLO4=^D9464                 ;LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
 4224          000001                    MAXHI4=1                      ;HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
 4225          002734                    RNDSHT=^D1500                 ;SHORT RANDOM SEEKS = 1500(10)
 4226          016514                    RNDLNG=^D7500                 ;LONG RANDOM SEEKS = 7500(10)
 4227          000002                    LSTTRK=2                      ;LAST TRACK  = 2
 4228          000365                    ALNCYL=365                    ;ALIGNMENT CYLINDER =245(10)
 4229          000002                    BSERR=BIT1                    ;BSE ERROR
 4230          000004                    HVRCER=BIT2                   ;HVRC ERROR
 4231          000010                    OPIERR=BIT3                   ;OPI ERROR
 4232          000020                    DCKERR=BIT4                   ;DATA CHECK ERROR
 4233          000040                    ECCNC=BIT5                    ;ECC NON-CORRECTABLE
 4234          000100                    WCERR=BIT6                    ;WRITE CHECK ERROR
 4235          000200                    ABORT=BIT7                    ;ABORT
 4236          000400                    LEV2ER=BIT8                   ;LEVEL TWO ERROR
 4237          001000                    BADSEC=BIT9                   ;BAD SECTOR FLAG
 4238          002000                    TWOTOS=BIT10                  ;TWO TIME OUTS
 4239          004000                    RCLREQ=BIT11                  ;RECALIBRATE REQUIRED
 4240          010000                    DRNAVL=BIT12                  ;DRIVE NOT RELSD BY OTHER PORT
 4241          100000                    ^NYDER=BIT15                  ;ANY ERROR DETECTED FLAG
 4242
 4243  007036  005015  055103  033122  DZR6N:  .ASCIZ  <15><12>/CZR6NE0/
 4244  007044  042516  000060
 4245  007050  051040  033113  030461  SUBVER: .ASCIZ  & RK611/06 SS VFY&
 4246  007056  030057  020066  051523
```

```
 4247  007064  053040  054506     000
 4248  007071     061  005015     000  PART1:  .ASCIZ  /1/<15><12>
 4249  007075     062  005015     000  PART2:  .ASCIZ  /2/<15><12>
 4250  007101     015  005012  040514  LSTMEM: .ASCIZ  <15><12><12>/LAST PHYS MEM ADR = /
 4251  007106  052123  050040  054510
 4252  007114  020123  042515  020115
 4253  007122  042101  020122  020075
 4254  007130     000
 4255  007131     117  042526  046122  OVLODR: .ASCIZ  /OVERLAY LOADER ? (Y OR N) * /
 4256  007136  054501  046040  040517
 4257  007144  042504  020122  020077
 4258  007152  054450  047440  020122
 4259  007160  024516  025040  000040
 4260  007166  042522  047515  042526  XXDPME: .ASCII /REMOVE XXDP PACK FROM DRIVE, INSTALL SCRATCH /<15><12><377>
 4261  007174  054040  042130  020120
 4262  007202  040520  045503  043040
 4263  007210  047522  020115  051104
 4264  007216  053111  026105  044440
 4265  007224  051516  040524  046114
 4266  007232  051440  051103  052101
 4267  007240  044103  006440  177412
 4268  007246  040520  045503  020054          .ASCIZ /PACK, AND HIT CONTINUE TO PROCEED./
 4269  007254  047101  020104  044510
 4270  007262  020124  047503  052116
 4271  007270  047111  042525  052040
 4272  007276  020117  051120  041517
 4273  007304  042505  027104     000
 4274  007311     015  050012  051101  PRMINP: .ASCIZ  <15><12>/PARAM INPUT MODE/<15><12><12>
 4275  007316  046501  044440  050116
 4276  007324  052125  046440  042117
 4277  007332  006505  005012     000
 4278  007337     122  030113  026466  RKBADR: .ASCIZ  /RK06-07 BUS ADR = /
 4279  007344  033460  041040  051525
 4280  007352  040440  051104  036440
 4281  007360  000040
 4282  007362  045522  033060  030055  RKVADR: .ASCIZ  /RK06-07 VEC ADR = /
 4283  007370  020067  042526  020103
 4284  007376  042101  020122  020075
 4285  007404     000
 4286  007405     122  030113  026466  RKPRTY: .ASCIZ  /RK06-07 PRIOR =/
 4287  007412  033460  050040  044522
 4288  007420  051117  036440     000
 4289  007425     123  051127  036440  SWRMSG: .ASCIZ  /SWR = /
 4290  007432  000040
 4291  007434  020040  042516  020127  NEWMSG: .ASCIZ  /  NEW = /
 4292  007442  020075     000
 4293  007445     015  042012  053122  DRVSEQ: .ASCIZ  <15><12>/DRV(S) = /
 4294  007452  051450  020051  020075
 4295  007460     000
 4296  007461     104  044522  042526  BADDRV: .ASCIZ  /DRIVE   /
 4297  007466  020040  000040
 4298  007472  047516  026516  054105  NXDRIV: .ASCIZ  /NON-EXIST/<15><12>
 4299  007500  051511  006524  000012
 4300  007506  047516  020124  042122  NTREDY: .ASCIZ  /NOT RDY/<15><12>
 4301  007514  006531  000012
 4302  007520  051127  026524  047514  WRTLOK: .ASCIZ  /WRT-LOCK/<15><12>
```

```
 4303  007526  045503  005015     000
 4304  007533     114  040517  042504  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
 4305  007540  020104  044527  044124
 4306  007546  040440  044514  047107
 4307  007554  050040  041501  006513
 4308  007562  000012
 4309  007564  005015  025012  020052  NODRTS: .ASCII  <15><12><12>/** NO DRVS TO TEST/<15><12>
 4310  007572  047516  042040  053122
 4311  007600  020123  047524  052040
 4312  007606  051505  006524     012
 4313  007613     120  042522  051523  CNTRDY: .ASCIZ  /PRESS ''CONT'' WHEN RDY/<15><12>
 4314  007620  021040  047503  052116
 4315  007626  020042  044127  047105
 4316  007634  051040  054504  005015
 4317  007642     000
 4318  007643     110  046101  020124  HLTRQD: .ASCIZ  /HALT REQ/<15><12>
 4319  007650  042522  006521  000012
 4320  007656  005015  052012  020117  ALDRVS: .ASCIZ  <15><12><12>/TO TEST ALL DRVS TYPE ''A'' <CR>, ELSE <CR>/<15><12>/* /
 4321  007664  042524  052123  040440
 4322  007672  046114  042040  053122
 4323  007700  020123  054524  042520
 4324  007706  021040  021101  036040
 4325  007714  051103  026076  042440
 4326  007722  051514  020105  041474
 4327  007730  037122  005015  020052
 4328  007736     000
 4329  007737     015  046012  036440  TSTMDS: .ASCII  <15><12>/L = LIST TESTS/<15><12>
 4330  007744  046040  051511  020124
 4331  007752  042524  052123  006523
 4332  007760     012
 4333  007761     103  036440  041440          .ASCII  /C = CHANGE TESTS/<15><12>
 4334  007766  040510  043516  020105
 4335  007774  042524  052123  006523
 4336  010002     012
 4337  010003     111  036440  044440          .ASCIZ  /I = INPUT PARAMS, RUN TESTS/<15><12>
 4338  010010  050116  052125  050040
 4339  010016  051101  046501  026123
 4340  010024  051040  047125  052040
 4341  010032  051505  051524  005015
 4342  010040     000
 4343  010041     015  042412  052116  ENTLCI: .ASCIZ  <15><12>/ENTER L,C, OR I/<15><12>/* /
 4344  010046  051105  046040  041454
 4345  010054  020054  051117  044440
 4346  010062  005015  020052     000
 4347  010067     124  020117  042504  DFTEST: .ASCIZ  /TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12>/* /
 4348  010074  040506  046125  020124
 4349  010102  042524  052123  020123
 4350  010110  054524  042520  042040
 4351  010116  036040  051103  026076
 4352  010124  042440  051514  020105
 4353  010132  041474  037122  005015
 4354  010140  020052     000
 4355  010143     015  052012  051505  TLSTHD: .ASCIZ  <15><12>/TEST     ITERATIONS/<15><12>
 4356  010150  020124  020040  020040
 4357  010156  052111  051105  052101
 4358  010164  047511  051516  005015
```

```
 4359  010172     000
 4360  010173     015  052012  036440  EXPLAN: .ASCII  <15><12>/T = TYPE PARAM LIST/<15><12>
 4361  010200  052040  050131  020105
 4362  010206  040520  040522  020115
 4363  010214  044514  052123  005015
 4364  010222  020117  020075  050117          .ASCII  /O = OPEN PARAM LIST/<15><12>
 4365  010230  047105  050040  051101
 4366  010236  046501  046040  051511
 4367  010244  006524     012
 4368  010247     123  036440  051440          .ASCII  /S = SET INDIVIDUAL PARAM/<15><12>
 4369  010254  052105  044440  042116
 4370  010262  053111  042111  040525
 4371  010270  020114  040520  040522
 4372  010276  006515     012
 4373  010301     122  036440  051040          .ASCIZ  /R = RUN TESTS/<15><12>
 4374  010306  047125  052040  051505
 4375  010314  051524  005015     000
 4376  010321     015  042412  052116  PARMDE: .ASCIZ  <15><12>/ENTER T,O,S, OR R/<15><12>
 4377  010326  051105  052040  047454
 4378  010334  051454  020054  051117
 4379  010342  051040  005015     000
 4380  010347     015  025012  042040  DUACES: .ASCIZ  <15><12>/* DUAL-ACCESS DATA TEST */<15><12>
 4381  010354  040525  026514  041501
 4382  010362  042503  051523  042040
 4383  010370  052101  020101  042524
 4384  010376  052123  025040  005015
 4385  010404     000
 4386  010405     115  054101  053440  MAWRDC: .ASCIZ  /MAX WORD COUNT = /
 4387  010412  051117  020104  047503
 4388  010420  047125  020124  020075
 4389  010426     000
 4390  010427     052  020052  051440  SECNL1: .ASCII  /**  S0>S1/
 4391  010434  037060  030523
 4392  010440  047040  052117  040440  NOTALD: .ASCIZ  / NOT ALLOWED/<15><12>
 4393  010446  046114  053517  042105
 4394  010454  005015     000
 4395  010457     052  020052  051440  SECNL2: .ASCIZ  /**  S2>S3/
 4396  010464  037062  031523     000
 4397  010471     052  020052  043040  TRKNLW: .ASCIZ  /**  FT>LT/
 4398  010476  037124  052114     000
 4399  010503     052  020052  053440  WC2BIG: .ASCIZ  /**  WC OR MA TOO LARGE/<15><12>
 4400  010510  020103  051117  046440
 4401  010516  020101  047524  020117
 4402  010524  040514  043522  006505
 4403  010532  000012
 4404  010534  047524  042040  043105  DFQUES: .ASCIZ  /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR>/<15><12>/* /
 4405  010542  052501  052114  040440
 4406  010550  046114  050040  051101
 4407  010556  046501  026123  052040
 4408  010564  050131  020105  020104
 4409  010572  041474  037122  020054
 4410  010600  046105  042523  036040
 4411  010606  051103  006476  025012
 4412  010614  000040
 4413  010616  005015  051525  051105  PFIFTN: .ASCIZ  <15><12>/USER-DEF PATT 15 :/<15><12>
 4414  010624  042055  043105  050040
```

```
 4415  010632  052101  020124  032461
 4416  010640  035040  005015     000
 4417  010645     015  046412  042117  SELP15: .ASCIZ  <15><12>/MOD PATT 15 :/<15><12>
 4418  010652  050040  052101  020124
 4419  010660  032461  035040  005015
 4420  010666     000
 4421  010667     124  020117  047515  MDFY15: .ASCIZ  /TO MOD PATT 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
 4422  010674  020104  040520  052124
 4423  010702  030440  026065  052040
 4424  010710  050131  020105  020115
 4425  010716  041474  037122  020054
 4426  010724  046105  042523  036040
 4427  010732  051103  006476  025012
 4428  010740  000040
 4429  010742  005015  047105  042524  ENTPAS: .ASCIZ  <15><12>/ENTER NO. OF PASSES (1-77777) :/<15><12>/* /
 4430  010750  020122  047516  020056
 4431  010756  043117  050040  051501
 4432  010764  042523  020123  030450
 4433  010772  033455  033467  033467
 4434  011000  020051  006472  025012
 4435  011006  000040
 4436  011010  005015  025052  020040  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRV - 20 ERRORS/<15><12>
 4437  011016  051104  050117  044520
 4438  011024  043516  042040  053122
 4439  011032  026440  031040  020060
 4440  011040  051105  047522  051522
 4441  011046  005015     000
 4442  011051     015  005012  042524  TSTDRN: .ASCII  <15><12><12>/TESTING DRV /
 4443  011056  052123  047111  020107
 4444  011064  051104  020126
 4445  011070  006440  000012          DRVNO:  .ASCIZ  / /<15><12>
 4446  011074  051104  020126     000  DRIV:   .ASCIZ  /DRV /
 4447  011101     103  051101  027124  CART:   .ASCIZ  /CART. /
 4448  011106  000040
 4449  011110  042523  027122  047040  SERNM:  .ASCIZ  /SER. NO.  /
 4450  011116  027117  020040     000
 4451  011123     015  043012  041501  FACTBS: .ASCIZ  <15><12>/FACTORY /
 4452  011130  047524  054522  000040
 4453  011136  051412  043117  053524  SOFTBS: .ASCII  <12>/SOFTWARE /
 4454  011144  051101  020105
 4455  011150  040502  020104  042523  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
 4456  011156  052103  051117  020123
 4457  011164  006472  000012
 4458  011170  020040  047516  042516  NOFALS: .ASCIZ  /  NONE/
 4459  011176     000
 4460
 4461
 4462                                    ;MESSAGES USED IN OFFSET-TO-FAILURE TEST
 4463  011177     015  005012  043117  OFSHED: .ASCII  <15><12><12>/OFFSET-TO-FAILURE MEAS :/<15><12><12>
 4464  011204  051506  052105  052055
 4465  011212  026517  040506  046111
 4466  011220  051125  020105  042515
 4467  011226  051501  035040  005015
 4468  011234     012
 4469  011235     124  040522  045503          .ASCII  /TRACK  CYLN  SECT  +OFST  -OFST/<15><12>
 4470  011242  020040  054503  047114
```

```
 4471  011250  020040  042523  052103
 4472  011256  020040  047453  051506
 4473  011264  020124  026440  043117
 4474  011272  052123  005015
 4475  011276  004411  020040  024040          .ASCIZ  <011><011>/   (UIN)  (UIN)/<15><12>
 4476  011304  044525  024516  020040
 4477  011312  052450  047111  006451
 4478  011320  000012
 4479
 4480
 4481  011322     000                  UNLOD:  .BYTE   0               ;DRIVE UNLOADED INDICATOR
 4482  011323     000                  VERIFY: .BYTE   0               ;VERIFY MODE FLAG
 4483  011324  005015  025012  025052  IDENT:  .ASCIZ  <15><12><12>/*** RK06-07 HEAD ALIGN AID ***/<15><12>
 4484  011332  051040  030113  026466
 4485  011340  033460  044040  040505
 4486  011346  020104  046101  043511
 4487  011354  020116  044501  020104
 4488  011362  025052  006452  000012
 4489  011370  047506  020122  042510  FORHLP: .ASCIZ  /FOR HELP TYPE H, ELSE <CR>/<15><12>
 4490  011376  050114  052040  050131
 4491  011404  020105  026110  042440
 4492  011412  051514  020105  041474
 4493  011420  037122  005015     000
 4494  011425     012  040515  052516  TYPMOD: .ASCIZ  <12>/MANUAL OR AUTO MODE (M OR A)?/<15><12>
 4495  011432  046101  047440  020122
 4496  011440  052501  047524  046440
 4497  011446  042117  020105  046450
 4498  011454  047440  020122  024501
 4499  011462  006477  000012
 4500  011466  005012  020052  040515  TYPMAN: .ASCIZ  <12><12>/* MANUAL SELECT MODE */<15><12>
 4501  011474  052516  046101  051440
 4502  011502  046105  041505  020124
 4503  011510  047515  042504  025040
 4504  011516  005015     000
 4505  011521     012  047105  042524  ENTDRV: .ASCIZ  <12>/ENTER DRV NO. (0-7):/<15><12>
 4506  011526  020122  051104  020126
 4507  011534  047516  020056  030050
 4508  011542  033455  035051  005015
 4509  011550     000
 4510  011551     052  020052  042040  NOTRDY: .ASCII  /**  DRV /
 4511  011556  053122     040
 4512  011561     040  047040  052117  DRNRDY: .ASCII  /  NOT RDY !  START DRV IF NECESSARY./<15><12>
 4513  011566  051040  054504  020440
 4514  011574  020040  052123  051101
 4515  011602  020124  051104  020126
 4516  011610  043111  047040  041505
 4517  011616  051505  040523  054522
 4518  011624  006456     012
 4519  011627     040  020040  053440          .ASCIZ  /    WHEN DRV IS RDY, PRESS ''CONT'' ON CPU./<15><12>
 4520  011634  042510  020116  051104
 4521  011642  020126  051511  051040
 4522  011650  054504  020054  051120
 4523  011656  051505  020123  041442
 4524  011664  047117  021124  047440
 4525  011672  020116  050103  027125
 4526  011700  005015     000
```

```
 4527  011703     052  020052  042040  NONEXD: .ASCII  /**  DRV /
 4528  011710  053122     040
 4529  011713     040  047040  047117  DRVNED: .ASCIZ  /  NON-EXIST OR OFF-LINE ! PRESS ''CONT'' TO RESTART./<15><12>
 4530  011720  042455  044530  052123
 4531  011726  047440  020122  043117
 4532  011734  026506  044514  042516
 4533  011742  020440  050040  042522
 4534  011750  051523  021040  047503
 4535  011756  052116  020042  047524
 4536  011764  051040  051505  040524
 4537  011772  052122  006456  000012
 4538  012000  025052  020040  051104  NOTLOK: .ASCII  /**  DRV /
 4539  012006  020126
 4540  012010  020040  047516  020124  NODRLK: .ASCII  /  NOT WRT-LOCK ! PLEASE SET WRT-LOC SW ./<15><12>
 4541  012016  051127  026524  047514
 4542  012024  045503  020440  050040
 4543  012032  042514  051501  020105
 4544  012040  042523  020124  051127
 4545  012046  026524  047514  020103
 4546  012054  053523  027040  005015
 4547  012062  020040  020040  044124          .ASCIZ  /    THEN PRESS ''CONT'' ON CPU./<15><12>
 4548  012070  047105  050040  042522
 4549  012076  051523  021040  047503
 4550  012104  052116  020042  047117
 4551  012112  041440  052520  006456
 4552  012120  000012
 4553  012122  025052  020040  052515  MULSEL: .ASCIZ  /**  MULT DRV AUTO-SELECTED ! PRESS 'CONT' TO RESTART./<15><12>
 4554  012130  052114  042040  053122
 4555  012136  040440  052125  026517
 4556  012144  042523  042514  052103
 4557  012152  042105  020440  050040
 4558  012160  042522  051523  023440
 4559  012166  047503  052116  020047
 4560  012174  047524  051040  051505
 4561  012202  040524  052122  006456
 4562  012210  000012
 4563  012212  040412  044514  047107  ENTMOD: .ASCIZ  <12>/ALIGN,VERIFY, OR EXERCISE (A,V, OR E) ?/<15><12>
 4564  012220  053054  051105  043111
 4565  012226  026131  047440  020122
 4566  012234  054105  051105  044503
 4567  012242  042523  024040  026101
 4568  012250  026126  047440  020122
 4569  012256  024505  037440  005015
 4570  012264     000
 4571  012265     012  025012  051040  MANEXR: .ASCII  <12><12>/* RANDOM SEEK EXERCISES IN PROGRESS /
 4572  012272  047101  047504  020115
 4573  012300  042523  045505  042440
 4574  012306  042530  041522  051511
 4575  012314  051505  044440  020116
 4576  012322  051120  043517  042522
 4577  012330  051523     040
 4578  012333     117  020116  051104          .ASCII  /ON DRIVE /
 4579  012340  053111  020105
 4580  012344  020040  006452  000012  DRIEXR: .ASCIZ  /  */<15><12>
 4581  012352  005012  020052  040515  MANALN: .ASCIZ  <12><12>/* MANUAL SELECT ALIGNMENT */<15><12>
 4582  012360  052516  043101  051440
```

```
 4583  012366  046105  041505  020124
 4584  012374  046101  043511  046516
 4585  012402  047105  020124  006452
 4586  012410  000012
 4587  012412  005012  020052  040515  MANVRF: .ASCIZ  <12><12>/* MANUAL SELECT VERIFY */<15><12>
 4588  012420  052516  046101  051440
 4589  012426  046105  041505  020124
 4590  012434  042526  044522  054506
 4591  012442  025040  005015     000
 4592  012447     110  040505  051504  HEDPOS: .ASCIZ  /HEADS AT CYL 365 (OCT) (760 RK07)/<15><12>
 4593  012454  040440  020124  054503
 4594  012462  020114  033063  020065
 4595  012470  047450  052103  020051
 4596  012476  033450  030066  051040
 4597  012504  030113  024467  005015
 4598  012512     000
 4599  012513     105  052116  051105  DTYMSG: .ASCIZ  /ENTER DRIVE TYPE /
 4600  012520  042040  044522  042526
 4601  012526  052040  050131  020105
 4602  012534     000
 4603  012535     040  020040  020040  SOR7:   .ASCIZ  /     : <6> OR <7>/
 4604  012542  020072  033074  020076
 4605  012550  051117  036040  037067
 4606  012556     000
 4607  012557     012  047105  042524  ENTHED: .ASCIZ  <12>/ENTER HEAD NO. (0-2):/<15><12>
 4608  012564  020122  042510  042101
 4609  012572  047040  027117  024040
 4610  012600  026460  024462  006472
 4611  012606  000012
 4612  012610  054524  042520  036040  RWNRDY: .ASCIZ  /TYPE <R> WHEN RDY :/<15><12>
 4613  012616  037122  053440  042510
 4614  012624  020116  042122  020131
 4615  012632  006472  000012
 4616  012636  042510  042101     040  HEDSEL: .ASCII  /HEAD /
 4617  012643     040  051440  046105  HEADNO: .ASCIZ  /  SELECTED/<15><12>
 4618  012650  041505  042524  006504
 4619  012656  000012
 4620  012660  005012  020052  052501  TYPAUT: .ASCIZ  <12><12>/* AUTO SELECT MODE */<15><12>
 4621  012666  047524  051440  046105
 4622  012674  041505  020124  047515
 4623  012702  042504  025040  005015
 4624  012710     000
 4625  012711     012  025012  040440  AUTALN: .ASCIZ  <12><12>/* AUTO SELECT ALIGNMENT */<15><12>
 4626  012716  052125  020117  042523
 4627  012724  042514  052103  040440
 4628  012732  044514  047107  042515
 4629  012740  052116  025040  005015
 4630  012746     000
 4631  012747     012  025012  040440  AUTVRF: .ASCIZ  <12><12>/* AUTO SELECT VERIFY */<15><12>
 4632  012754  052125  020117  042523
 4633  012762  042514  052103  053040
 4634  012770  051105  043111  020131
 4635  012776  006452  000012
 4636  013002  042012  053122     040  DRISEL: .ASCII  <12>/DRV /
 4637  013007     040  051440  046105  DRVSEL: .ASCIZ  /  SELECTED/<15><12><12>
 4638  013014  041505  042524  006504
```

```
 4639  013022  005012     000
 4640  013025     012  025012  040440  AUTEXR: .ASCII  <12><12>/* AUTO SELECT EXERCISES */<15><12>
 4641  013032  052125  020117  042523
 4642  013040  042514  052103  042440
 4643  013046  042530  041522  051511
 4644  013054  051505  025040  005015
 4645  013062  051412  047510  052122          .ASCIZ  <12>/SHORT OR LONG (S OR L) ?/<15><12>
 4646  013070  047440  020122  047514
 4647  013076  043516  024040  020123
 4648  013104  051117  046040  020051
 4649  013112  006477  000012
 4650  013116  054105  051105  044503  AUTOEX: .ASCII  /EXERCISING DRV /
 4651  013124  044523  043516  042040
 4652  013132  053122     040
 4653  013135     040  005015     000  AUTODR: .ASCIZ  / /<15><12>
 4654
 4655  013141     052  020052  041440  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRACK!/<15><12>
 4656  013146  047101  047516  020124
 4657  013154  042522  042101  041040
 4658  013162  042101  051440  041505
 4659  013170  047524  020122  051124
 4660  013176  041501  020513  005015
 4661  013204     000
 4662  013205     136  006503  000012  CNTRLC: .ASCIZ  /^C/<15><12>
 4663  013212  055136  005015     000  CNTRLZ: .ASCIZ  /^Z/<15><12>
 4664  013217     136  006522  000012  CNTRLR: .ASCIZ  /^R/<15><12>
 4665  013224  052536  005015     000  CNTRLU: .ASCIZ  /^U/<15><12>
 4666  013231     136  006507  000012  CNTRLG: .ASCIZ  /^G/<15><12>
 4667  013236  005015  000012          CR2LF:  .ASCIZ  <15><12><12>
 4668  013242  000134                  BKSLSH: .ASCIZ  /\/
 4669  013244  000054                  COMMA:  .ASCIZ  /,/
 4670  013246  020040                  SPACE6: .ASCII  /  /
 4671  013250     040                  SPACE4: .ASCII  / /
 4672  013251     040                  SPACE3: .ASCII  / /
 4673  013252     040                  SPACE2: .ASCII  / /
 4674  013253     040     000          SPACE1: .ASCIZ  / /
 4675          013256                          .EVEN
 4676  013256  020040  000075          PRMBUF: .ASCIZ  /  =/
 4677  013262  047527  042122  000040  WORDSP: .ASCIZ  /WORD /
 4678  013270  036440  000040          EQUALS: .ASCIZ  / = /
 4679  013274  020052     000          PROMPT: .ASCIZ  /* /
 4680  013277     076  000040          PRMPSP: .ASCIZ  /> /
 4681
 4682                                          .EVEN
 4683
 4684
 4685
 4686
 4687  013302  105037  003116          DFSTRT: CLRB    MDFLAG          ;SET FLAG FOR DEFAULT MODE
 4688  013306  105037  003135                  CLRB    DULACS          ;CLEAR DUAL-ACCESS FLAG
 4689  013312  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
 4690  013316  022737  014374  000042          CMP     #DRVTST,@#42    ;SEE IF EOP RETURN ADRS = DRVTST
 4691  013324  001003                          BNE     4$              ;BR IF NOT DRVTST
 4692  013326  012737  017260  000042          MOV     #NEWPAS,@#42    ;SET RETURN ADRS = NEWPAS
 4693  013334  000405                  4$:     BR      CMSTRT          ;PROCEED
 4694
```

```
 4695
 4696  013336  112737  000001  003116  PSTART: MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
 4697  013344  105037  003135                  CLRB    DULACS          ;CLEAR DUAL-ACCESS TEST FLAG
 4698
 4699  013350  012737  000340  177776  CMSTRT: MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
 4700                                  .SBTTL  INITIALIZE THE COMMON TAGS
 4701                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 4702  013356  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 4703  013362  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 4704  013364  022706  001140                  CMP     #SWR,R6 ;;DONE?
 4705  013370  001374                          BNE     .-6             ;;LOOP BACK IF NO
 4706  013372  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 4707                                  ;;INITIALIZE A FEW VECTORS
 4708  013376  012737  057146  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 4709  013404  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 4710  013412  012737  056442  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 4711  013420  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 4712  013426  012737  060254  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 4713  013434  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 4714  013442  012737  057012  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 4715  013450  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 4716  013456  013737  023754  023746          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 4717  013464  005037  001304                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 4718  013470  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 4719  013474  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 4720  013502  012737  013502  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 4721  013510  012737  013510  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 4722                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 4723                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 4724  013516  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 4725  013522  012737  013556  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 4726  01353C  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 4727  013536  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 4728  013544  022777  177777  165366          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 4729  013552  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 4730                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
 4731  013554  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
 4732  013556  012716  013564          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 4733  013562  000002                          RTI
 4734  013564  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 4735  013572  012737  000174  001142          MOV     #DISPREG,DISPLAY
 4736  013600  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 4737
 4738  013604  005037  001326                  CLR     $PASS           ;;CLEAR PASS COUNT
 4739  013610  132737  000200  001341          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
 4740  013616  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
 4741  013620  012737  001342  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
 4742  013626                          67$:
 4743  013626  005037  003150                  CLR     AUTOFG          ;CLEAR THE SPECIAL FLAG
 4744  013632  000005                          RESET                   ;CLEAR THE UNIBUS
 4745  013634  012737  000006  000004          MOV     #6,@#ERRVEC     ;SET TIME-OUT VECTOR
 4746  013642  005037  000006                  CLR     @#ERRVEC+2
 4747  013646  012737  031366  000114          MOV     #MPERHD,@#MEMVEC ;SET MEM PARITY TRAP VECTOR
 4748  013654  012737  000340  000116          MOV     #PR7,@#MEMVEC+2 ;SET TRAP PRIORITY = 7
 4749  013662  004737  031302                  JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
 4750  013666  112737  000144  001115          MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 F - $SCOPE
```

```
 4751  013674  105037  003130                    CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
 4752  013700  112737  000001  003145            MOVB    #1,HLPOVL       ;SET HLP FILE OVLD INDICTR
 4753  013706  104401  007036                    TYPE    ,DZR6N          ;TYPE PROGRAM I.D. FOR PART 2
 4754  013712  104401  007050                    TYPE    ,SUBVER
 4755  013716  104401  007075                    TYPE    ,PART2
 4756  013722  105737  003146                    TSTB    NOTYPE          ;SEE IF OPERATOR NOTE SHOULD BE TYPED
 4757  013726  001004                            BNE     39$             ;BR IF NOT
 4758  013730  104401  065636                    TYPE    ,NOTMSG         ;TYPE OPERATOR NOTE
 4759  013734  105237  003146                    INCB    NOTYPE          ;SET FLAG FOR NEXT TIME
 4760  013740  105737  003135          39$:      TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
 4761  013744  001402                            BEQ     40$             ;BR IF NOT
 4762  013746  104401  010347                    TYPE    ,DUACES         ;TYPE "* DUAL-ACCESS DATA TEST *"
 4763  013752  012737  030376  000060  40$:      MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
 4764  013760  012737  000200  000062            MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
 4765  013766  013701  003040                    MOV     RKVEC,R1        ;ADDR. OF RK06 VECTOR STORAGE
 4766  013772  012721  047262                    MOV     #I.INTR,(R1)+   ;SET IT TO RK06 HANDLER
 4767  013776  013711  003042                    MOV     RKPRI,(R1)      ;SET RK06 PRIORITY
 4768  014002  012737  031206  000250            MOV     #KTERHD,@#MMVEC ;VECT FOR KT11 FAILURE
 4769  014010  012737  000340  000252            MOV     #PR7,@#MMVEC+2  ;SET PRIOR. = 7 FOR HNDLER
 4770  014016  105037  003120                    CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
 4771  014022  012737  000000  177776            MOV     #PR0,@#PS       ;ALLOW ALL INTERRUPTS AGAIN
 4772  014030  012701  00522?                    MOV     #PRVCMD,R1      ;ZERO OUT PREVIOUS COMMAND
 4773  014034  012700  000006                    MOV     #6,R0           ;SET COUNTER
 4774  014040  005021                  42$:      CLR     (R1)+           ;ZERO A WORD
 4775  014042  005300                            DEC     R0
 4776  014044  001375                            BNE     42$             ;BR IF NOT DONE YET
 4777  014046  005037  005502                    CLR     STALLS          ;DON'T ALLOW STALLS YET
 4778  014052  023727  000042  017260            CMP     @#42,#NEWPAS    ;SEE IF CHAIN MODE
 4779  014060  101002                            BHI     44$             ;BR IF YES
 4780  014062  004737  031066                    JSR     PC,GTSWRG       ;OPE   OFTWARE SWR FOR MODIFICATION
 4781  014066  012737  176543  054710  44$:      MOV     #176543,$HINUM  ;INI   PSEUDO-RANDOM NOS.
 4782  014074  012737  123456  054712            MOV     #123456,$LONUM
 4783  014102  004737  030570                    JSR     PC,SIZMEM       ;SIZE MEMORY, FIX MA LIMIT IN PRMLIM
 4784  014106  105737  003134                    TSTB    XDPSVD          ;SEE IF XXDP PREVIOUSLY SAVED
 4785  014112  001404                            BEQ     9$              ;BR IF NOT
 4786  014114  004737  032010                    JSR     PC,GETXDP       ;RESTORE SAVED XXDP
 4787  014120  105037  003134                    CLRB    XDPSVD          ;CLEAR THE FLAG
 4788  014124  105737  003116          9$:       TSTB    MDFLAG          ;SEE IF DEFAULT MODE
 4789  014130  001031                            BNE     10$             ;BR IF NOT DEFAULT MODE
 4790  014132  013700  001370                    MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
 4791  014136  013737  001374  003036            MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
 4792  014144  132737  000200  001341            BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
 4793  014152  001005                            BNE     18$             ;BR IF NO SIZING
 4794  014154  012700  120210                    MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
 4795  014160  012737  177440  003036            MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
 4796  014166  110037  003040          18$:      MOVB    R0,RKVEC        ;STORE VECTOR
 4797  014172  105037  003041                    CLRB    RKVEC+1         ;CLEAR HI BYTE
 4798  014176  000300                            SWAB    R0              ;GET PRTY INTO BITS 5-7
 4799  014200  042700  177437                    BIC     #177437,R0      ;CLEAR OTHER BITS
 4800  014204  010037  003042                    MOV     R0,RKPRI        ;STORE RK06 PRIORITY
 4801  014210  000137  015060                    JMP     ALLDRV          ;GO CHECK ALL DRIVES
 4802
 4803
 4804
 4805                                    ;BEGIN PARAMETER INPUT MODE
 4806  014214  104401  007311          10$:      TYPE    ,PRMINP         ;TYPE "PARAMETER INPUT MODE"
```

```
                                ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
014220  013746  003036                  MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
014224  104401  007337                  TYPE    ,RKBADR         ;TYPE 'RK06 BUS ADR = ''
014230  004737  030756                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
014234  012637  003036                  MOV     (SP)+,RKBAS     ;STORE NEW VALUE
                                ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
014240  013746  003040                  MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
014244  104401  007362                  TYPE    ,RKVADR         ;TYPE 'RK06 VEC ADR = ''
014250  004737  030756                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
014254  012637  003040                  MOV     (SP)+,RKVEC     ;STORE NEW VALUE
                                ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
014260  013746  003042          11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
014264  006316                  ASL     (SP)            ;GET IT INTO BITS 0-2
014266  006316                  ASL     (SP)
014270  006316                  ASL     (SP)
014272  000316                  SWAB    (SP)
014274  104401  007405                  TYPE    ,RKPRTY         ;TYPE   'RK06 PRIORITY = ''
014300  004737  030756                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
014304  012600                  MOV     (SP)+,R0
014306  020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
014312  002414                  BLT     12$             ;BR IF NEW VALUE TOO SMALL
014314  020027  000007                  CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
014320  003011                  BGT     12$             ;BR IF NEW VALUE TOO LARGE
014322  000300                  SWAB    R0              ;GET PRIORITY INTO BITS 5-7
014324  006200                  ASR     R0
014326  006200                  ASR     R0
014330  006200                  ASR     R0
014332  010037  003042                  MOV     R0,RKPRI        ;STORE NEW VALUE
014336  104401  001315                  TYPE    ,$CRLF
014342  000405                  BR      16$
014344  104401  005264          12$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
014350  104401  001314                  TYPE    ,$QUES
014354  000741                  BR      11$             ;GO ASK AGAIN
014356  105037  003126          16$:    CLRB    ERRCNT          ;CLEAR ERROR CNT FOR RESTARTS
014362  012737  014374  000042          MOV     #DRVTST,@#42    ;SET UP DUMP MODE RETURN FROM $EOP
                                ;       UPON COMPLETION OF REQUESTED PASSES
014370  000137  015136                  JMP     CHKLST          ;GO CHECK STATUS OF MARKED DRIVES


                                ;;*****************************************************************
                                .SBTTL  TO - DESIRED DRIVE INPUT ROUTINE
                                ;*THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
                                ;*WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
                                ;*CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
                                ;*DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
                                ;*AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
                                ;*LIST (DRVLST).
                                ;;*****************************************************************

014374                          DRVTST:
014374  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK
014400  005037  005502                  CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
014404  004737  031302                  JSR     PC, WBCSR       ;ENABLE MEMORY PARITY CHECK
014410  104401  007656                  TYPE    ,ALDRVS         ;ASK IF ALL DRIVES DESIRED
```

```
4863  014414  004737  034002              JSR     PC,RDCHRS       ;READ RESPONSE
4864  014420  014374                      DRVTST                  ;(^C) RETURN ADDRESS
4865  014422  014374                      DRVTST                  ;(^Z) RETURN ADDRESS
4866  014424  014374                      DRVTST                  ;(^U) OR ERROR RETURN ADDRESS
4867  014426  005700                      TST     R0              ;SEE IF NULL INPUT
4868  014430  001413                      BEQ     TELDRV          ;BR IF NULL INPUT
4869  014432  022737  000101  005264      CMP     #'A,BUFF0       ;SEE IF ALL DRIVES REQUESTED
4870  014440  001002                      BNE     4$              ;BR IF NOT ALL DRIVES
4871  014442  000137  015060              JMP     ALLDRV          ;CHECK ALL DRIVES
4872  014446  104401  005264      4$:     TYPE    ,BUFF0          ;ECHO BAD INPUT
4873  014452  104401  001314              TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
4874  014456  000746                      BR      DRVTST          ;GO ASK AGAIN
4875                              ;TYPE CURRENT DRIVE LIST
4876  014460  104401  007445      TELDRV: TYPE    ,DRVSEQ         ;TYPE 'DRIVE(S) = ''
4877  014464  005001                      CLR     R1              ;INITIALIZE DRIVE NUMBER
4878  014466  005000                      CLR     R0              ;INIT. COUNT OF LISTED DRIVES
4879  014470  012703  000001              MOV     #BIT0,R3        ;INIT BIT POINTER
4880  014474  105761  005610      4$:     TSTB    DRVLST(R1)      ;SEE IF THIS DRIVE IS LISTED
4881  014500  001414                      BEQ     8$              ;BR IF DRIVE NOT LISTED
4882  014502  005700                      TST     R0              ;SEE IF FIRST TIME HERE
4883  014504  001402                      BEQ     6$              ;BR IF FIRST TIME HERE
4884  014506  104401  013244              TYPE    ,COMMA          ;TYPE A COMMA
4885  014512  010137  005534      6$:     MOV     R1,SCRACH       ;USE SCRACH FOR BUFFER
4886  014516  052737  000060  005534      BIS     #'0,SCRACH      ;CONVERT DRIVE NO. TO ASCII
4887  014524  104401  005534              TYPE    ,SCRACH         ;TYPE DRIVE NO.
4888  014530  005200                      INC     R0              ;INCREMENT NO. OF LISTED DRIVES
4889  014532  005201              8$:     INC     R1              ;INCREMENT DRIVE NO.
4890  014534  006303                      ASL     R3              ;SHIFT BIT POINTER
4891  014536  022701  000010              CMP     #10,R1          ;SEE IF DONE YET
4892  014542  001354                      BNE     4$              ;BR IF NOT DONE
4893  014544  005700                      TST     R0              ;SEE IF ANY DRIVES WERE LISTED
4894  014546  001016                      BNE     26$             ;BR IF YES
4895  014550  104401  011170              TYPE    ,NOFALS         ;TYPE ''  NONE''
4896  014554  104401  007564              TYPE    ,NODRTS         ;TYPE ''** NO DRIVES TO TEST''
4897                                                              ;     ''PRESS 'CONT' WHEN RDY''
4898  014560  012703  000401              MOV     #401,R3         ;PATTERN TO MARK DRIVES
4899  014564  012701  005610              MOV     #DRVLST,R1      ;DRIVE LIST ADDRESS
4900  014570  010321                      MOV     R3,(R1)+        ;MARK ALL DRIVES IN LIST
4901  014572  010321                      MOV     R3,(R1)+
4902  014574  010321                      MOV     R3,(R1)+
4903  014576  010311                      MOV     R3,(R1)
4904  014600  000137  046126              JMP     HLTPRG          ;HALT PROGRAM
4905  014604  104401  001315      26$:    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
4906  014610  122737  000013  000041      CMPB    #13,@#41        ;DID DRIVE LOAD XXDP??
4907  014616  001003                      BNE     28$             ;NO, SO SKIP NEXT
4908  014620  104401  007166              TYPE    ,XXDPME         ;INFORM USER TO REMOVE XXDP PACK
4909  014624  000000                      HALT                    ;AND FORCE OPERATOR INTERVENTION
4910  014626  105037  000041      28$:    CLRB    @#41            ;OPERATOR HIT CONT. CLR THE LD PTR
4911  014632  105737  003116              TSTB    MDFLAG          ;SEE IF DEFAULT MODE
4912  014636  001012                      BNE     19$             ;BR IF NOT DEFAULT MODE
4913  014640  000137  015476              JMP     LODFLS          ;GO LOAD DEFLT ITER. COUNTS
4914  014644  001007                      BNE     19$             ;BR IF NOT
4915  014646  105737  005610              TSTB    DRVLST          ;SEE IF DRIVE 0 LISTED TO TEST
4916  014652  001404                      BEQ     19$             ;BR IF NOT
4917  014654  004737  032250              JSR     PC,INITSS       ;INIT THE SUB-SYS
4918  014660  000137  015136              JMP     CHKLST          ;GO CHECK STATUS OF LISTED DRIVES
```

```
4919  014664  104401  013274          19$:    TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
4920                                  ;READ AND CHECK NEW DRIVE NUMBERS
4921  014670  004737  034002                  JSR     PC,RDCHRS       ;READ IN INPUT STRING
4922  014674  014374                          DRVTST                  ;(^C) RETURN ADDRESS
4923  014676  014374                          DRVTST                  ;(^Z) RETURN ADDRESS
4924  014700  014460                          TELDRV                  ;(^U) OR ERROR RETURN ADDRESS
4925  014702  005700                          TST     R0              ;SEE IF NULL INPUT
4926  014704  001007                          BNE     9$              ;BR IF NEW DRIVES WILL BE SELECTED
4927  014706  105737  003135                  TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
4928  014712  001002                          BNE     22$             ;BR IF YES
4929  014714  000137  015210                  JMP     ASKTST          ;JUMP TO THE TEST INPUT ROUTINE
4930  014720  000137  015754          22$:    JMP     INPUTP          ;GO ASK FOR INPUT PARAMETERS
4931  014724  022700  000017          9$:     CMP     #15.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
4932  014730  002005                          BGE     12$             ;BR IF NOT TOO MANY
4933  014732  104401  005264          10$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
4934  014736  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
4935  014742  000646                          BR      TELDRV          ;GO TYPE DRIVES AND ASK AGAIN
4936  014744  005001          12$:    CLR     R1                      ;CLEAR CHAR. POINTER
4937  014746  126127  005264  000060  14$:    CMPB    BUFF0(R1),#'0   ;SEE IF DRIVE NO. < 0
4938  014754  002766                          BLT     10$             ;BR TO ECHO BAD INPUT
4939  014756  126127  005264  000067          CMPB    BUFF0(R1),#'7   ;SEE IF > 7
4940  014764  003362                          BGT     10$             ;BR TO ECHO BAD INPUT
4941  014766  005201                          INC     R1              ;INCREMENT CHAR POINTER
4942  014770  020100                          CMP     R1,R0           ;SEE IF MORE CHARS TO CHECK
4943  014772  001406                          BEQ     16$             ;BR IF ALL CHARS CHECKED
4944  014774  126127  005264  000054          CMPB    BUFF0(R1),#',   ;SEE IF THIS IS COMMA
4945  015002  001353                          BNE     10$             ;BR IF NOT COMMA
4946  015004  005201                          INC     R1              ;INCREMENT CHAR POINTER
4947  015006  000757                          BR      14$             ;BR TO CONTINUE CHECKING CHARS
4948                                  ;GET NEW DRIVE NUMBERS INTO LIST
4949  015010  012701  005610          16$:    MOV     #DRVLST,R1      ;GET DRIVE LIST ADDRESS
4950  015014  005021                          CLR     (R1)+           ;CLEAR OUT THE DRIVE LIST
4951  015016  005021                          CLR     (R1)+
4952  015020  005021                          CLR     (R1)+
4953  015022  005011                          CLR     (R1)
4954  015024  005000                          CLR     R0              ;CLEAR BUFFER POINTER
4955  015026  116001  005264          18$:    MOVB    BUFF0(R0),R1    ;GET A DRIVE NUMBER
4956  015032  042701  000060                  BIC     #'0,R1          ;STRIP ASCII BITS
4957  015036  112761  000001  005610          MOVB    #1,DRVLST(R1)   ;MARK THIS DRIVE IN DRIVE LIST
4958  015044  062700  000002                  ADD     #2,R0           ;POINT TO NEXT DRIVE NUMBER
4959  015050  105760  005263                  TSTB    BUFF0-1(R0)     ;SEE IF NULL CHAR YET
4960  015054  001364                          BNE     18$             ;BR IF NOT DONE YET
4961  015056  000427                          BR      CHKLST          ;GO CHECK NEW DRIVES FOR VALID STATUS
4962                                  ;COME HERE IF ALL DRIVES REQUESTED
4963  015060  005000          ALLDRV: CLR     R0                      ;CLEAR DRIVE NUMBER
4964  015062  013703  001376                  MOV     $DEVM,R3        ;GET APT DEVICE MAP
4965  015066  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
4966  015074  001002                          BNE     1$              ;BR IF NO SIZING
4967  015076  012703  000377                  MOV     #377,R3         ;SET UP FOR SIZING
4968  015102  012701  000001          1$:     MOV     #BIT0,R1        ;SET BIT POINTER
4969  015106  105060  005610          2$:     CLRB    DRVLST(R0)      ;INITIALIZE DRIVE ENTRY
4970  015112  030103                          BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
4971  015114  001403                          BEQ     4$              ;BR IF NOT
4972  015116  112760  000001  005610          MOVB    #1,DRVLST(R0)   ;MARK THIS DRIVE IN DRIVE LIST
4973  015124  006301          4$:     ASL     R1                      ;SHIFT BIT POINTER
4974  015126  005200                          INC     R0              ;INCREMENT DRIVE NUMBER
```

```
 4975  015130  022700  000010                CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
 4976  015134  001364                        BNE     2$              ;BR IF NOT DONE YET
 4977                              ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
 4978  015136  005000              CHKLST: CLR     R0              ;CLEAR DRIVE NUMBER
 4979  015140  105760  005610      2$:     TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
 4980  015144  001410                        BEQ     4$              ;BR IF NOT MARKED
 4981  015146  010037  005500                MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
 4982  015152  004737  032404                JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE
 4983                                                                ;IF NOT VALID, TYPE MESSAGE
 4984                                                                ;   AND REMOVE IT FROM DRIVE LIST
 4985  015156  015202                        6$                      ;ERROR RETURN ADDRESS FOR SCNDRV
 4986  015160  113760  003124  005620        MOVB    TYPFMT,DTYLST(R0)       ;SET DRV TYP, 0=RK06, 4=RK07
 4987  015166  005200              4$:     INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
 4988  015170  022700  000010                CMP     #10,R0          ;SEE IF DONE CHECKING LIST
 4989  015174  001361                        BNE     2$              ;BR IF NOT DONE YET
 4990  015176  000137  014460                JMP     TELDRV          ;GO BACK TO LIST SELECTED DRIVES
 4991  015202  105060  005610      6$:     CLRB    DRVLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
 4992  015206  000767                        BR      4$              ;CONTINUE CHECKING THE LIST
 4993
 4994
 4995                              ;;*****************************************************************
 4996                              .SBTTL  TO - DESIRED TEST INPUT ROUTINE
 4997                              ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
 4998                              ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
 4999                              ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
 5000                              ;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
 5001                              ;*THAT THE TEST IS NOT TO BE RUN.
 5002                              ;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
 5003                              ;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
 5004                              ;*TYPED, (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
 5005                              ;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
 5006                              ;*OF OPERATING PARAMETERS, AND RUN TESTS.
 5007                              ;*TYPING (^C) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
 5008                              ;* (^Z) CAUSES RETURN TO ASKTMD.
 5009                              ;;*****************************************************************
 5010
 5011                              ;DETERMINE L,C, OR I MODE
 5012  015210  104401  007737      ASKTST: TYPE    ,TSTMDS         ;ASK FOR TEST INPUT MODE
 5013  015214  104401  010041      ASKTMD: TYPE    ,ENTLCI         ;TYPE "ENTER L,C, OR I"
 5014  015220  004737  034002              JSR     PC,RDCHRS       ;READ RESPONSE
 5015  015224  014374                      DRVTST                  ;(^C) RETURN ADDRESS
 5016  015226  015214                      ASKTMD                  ;(^Z) RETURN ADDRESS
 5017  015230  015214                      ASKTMD                  ;(^U) OR ERROR RETUP ADDRESS
 5018  015232  005700                      TST     R0              ;SEE IF ANY INPUT
 5019  015234  001005                      BNE     4$              ;BR IF ANY INPUT
 5020  015236  104401  005264      2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
 5021  015242  104401  001314              TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 5022  015246  000762                      BR      ASKTMD          ;GO ASK AGAIN
 5023  015250  022737  000114  005264  4$: CMP     #'L,BUFFO       ;SEE IF (L) TYPED
 5024  015256  001002                      BNE     6$              ;BR IF NOT (L)
 5025  015260  000137  015314              JMP     LSTTST          ;JUMP TO LIST TESTS
 5026  015264  022737  000103  005264  6$: CMP     #'C,BUFFO       ;SEE IF (C) TYPED
 5027  015272  001002                      BNE     8$              ;BR IF NOT (C)
 5028  015274  000137  015432              JMP     CHGTST          ;JUMP TO CHANGE TESTS
 5029  015300  022737  000111  005264  8$: CMP     #'I,BUFFO       ;SEE IF (I) TYPED
 5030  015306  001353                      BNE     2$              ;BR IF NOT (I), TO ECHO BAD INPUT
```

```
5031  015310  000137  015754                    JMP     INPUTP          ;GO INPUT PARAMS AND RUN TESTS
5032
5033                                    ;LIST CURRENT TESTS AND ITERATION COUNTS
5034  015314  104401  010143            LSTTST: TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
5035  015320  005001                            CLR     R1              ;INITIALIZE TEST INDEX
5036  015322  004737  030536            2$:     JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5037  015326  004737  034314                    JSR     PC,TYPTST       ;TYPE CURRENT TEST AND ITERATION NUMBER
5038  015332  104401  001315                    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5039  015336  005737  005522                    TST     INTCHR          ;SEE IF ANY INPUT
5040  015342  001416                            BEQ     8$              ;BR IF NO INPUT
5041  015344  122737  000003  005522            CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
5042  015352  001002                            BNE     4$              ;BR IF NOT (^C)
5043  015354  000137  014374                    JMP     DRVTST          ;JUMP TO ASK FOR DRIVES AGAIN
5044  015360  122737  000032  005522  4$:       CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
5045  015366  001002                            BNE     6$              ;BR IF NOT (^Z)
5046  015370  000137  015214                    JMP     ASKTMD          ;JUMP TO ASK FOR NEW TEST INPUT MODE
5047  015374  004737  030556            6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
5048  015400  062701  000002            8$:     ADD     #2,R1           ;INCREMENT TEST INDEX
5049  015404  010102                            MOV     R1,R2           ;GET COPY OF INDEX
5050  015406  062702  005630                    ADD     #TSTLST,R2      ;GET POSITION IN LIST
5051  015412  022702  005644                    CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
5052  015416  001341                            BNE     2$              ;BR IF NOT DONE YET
5053  015420  042777  000100  163516            BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
5054  015426  000137  015214                    JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
5055
5056                                    ;CHANGE CURRENT TESTS AND ITERATION COUNTS
5057  015432  104401  010067            CHGTST: TYPE    ,DFTEST         ;ASK IF TESTS SHOULD BE DEFAULTED
5058  015436  004737  034002                    JSR     PC,RDCHRS       ;READ RESPONSE
5059  015442  014374                            DRVTST                  ;(^C) RETURN ADDRESS
5060  015444  015214                            ASKTMD                  ;(^Z) RETURN ADDRESS
5061  015446  015432                            CHGTST                  ;(^U) OR ERROR RETURN ADDRESS
5062  015450  005700                            TST     R0              ;SEE IF NULL INPUT
5063  015452  001426                            BEQ     NULINP          ;BR IF NULL INPUT
5064  015454  022737  000104  005264            CMP     #'D,BUFFO       ;SEE IF (D) TYPED
5065  015462  001405                            BEQ     LODFLS          ;BR IF DEFAULTS REQUESTED
5066  015464  104401  005264                    TYPE    ,BUFFO          ;ECHO BAD INPUT
5067  015470  104401  001314                    TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5068  015474  000756                            BR      CHGTST          ;GO ASK AGAIN
5069                                  ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
5070  015476  012700  005644            LODFLS: MOV     #DFLTST,R0      ;LOAD DEFAULT LIST ADDRESS
5071  015502  012702  005630                    MOV     #TSTLST,R2      ;LOAD TEST LIST ADDRESS
5072  015506  012022                    4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
5073  015510  022702  005644                    CMP     #DFLTST,R2      ;SEE IF DONE YET
5074  015514  001374                            BNE     4$              ;BR IF NOT DONE YET
5075  015516  105737  003116                    TSTB    MDFLAG          ;SEE IF DEFAULT MODE
5076  015522  001234                            BNE     ASKTMD          ;BR IF NOT DEFAULT MODE
5077  015524  000137  016510                    JMP     LODFPT          ;GO LOAD DEFAULT PARAMETERS
5078  015530  005001            NULINP: CLR     R1              ;INITIALIZE TEST INDEX
5079  015532  104401  010143                    TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
5080                                  ;TYPE CURRENT TEST AND ITERATION NUMBER
5081  015536  004737  034314            8$:     JSR     PC,TYPTST       ;TYPE A TEST NO. AND ITERATION NO.
5082  015542  104401  013253                    TYPE    ,SPACE1         ;TYPE A SPACE
5083  015546  104401  013274                    TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
5084                                  ;READ AND CHECK INPUT, IF ANY
5085  015552  004737  034002                    JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
5086  015556  014374                            DRVTST                  ;(^C) RETURN ADDRESS
```

```
 5087  015560  015214                        ASKTMD                  ;(^Z) RETURN ADDRESS
 5088  015562  015536                        8$                      ;(^U) OR ERROR RETURN ADDRESS
 5089  015564  005700                        TST     R0              ;SEE IF ANY INPUT
 5090  015566  001012                        BNE     12$             ;BR IF ANY INPUT
 5091  015570  062701  000002          10$:  ADD     #2,R1           ;INCREMENT INDEX
 5092  015574  010102                        MOV     R1,R2           ;COPY INDEX
 5093  015576  062702  005630                ADD     #TSTLST,R2      ;GET POSITION IN LIST
 5094  015602  022702  005644                CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
 5095  015606  001353                        BNE     8$              ;BR IF NOT DONE YET
 5096  015610  000137  015214                JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
 5097  015614  022737  000041  005264  12$:  CMP     #'!,BUFFO       ;SEE IF (!) TYPED
 5098  015622  001431                        BEQ     16$             ;BR TO PROPAGATE CURRENT ITER. NO.
 5099  015624  005002                        CLR     R2              ;INITIALIZE (!) INDICATOR
 5100  015626  122760  000041  005263        CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
 5101  015634  001004                        BNE     14$             ;BR IF NOT (!)
 5102  015636  105060  005263                CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
 5103  015642  005300                        DEC     R0              ;DECREMENT CHAR COUNT
 5104  015644  005202                        INC     R2              ;SET (!) INDICATOR
 5105  015646  022700  000005          14$:  CMP     #5,R0           ;SEE HOW MANY CHARS NO.
 5106  015652  002433                        BLT     20$             ;BR IF TOO MANY (MAX ITER. NO. = 77776)
 5107  015654  012746  005264                MOV     #BUFFO,-(SP)    ;GET BUF ADDR. ON STACK FOR OCTBIN
 5108  015660  004737  054456                JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
 5109  015664  015742                        20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
 5110  015666  012600                        MOV     (SP)+,R0        ;GET ITERATION NUMBER
 5111  015670  020027  077776                CMP     R0,#77776       ;SEE IF > 77776
 5112  015674  101022                        BHI     20$             ;BR IF TOO BIG
 5113  015676  010061  005630                MOV     R0,TSTLST(R1)   ;PUT ITERATION NUMBER INTO TEST LIST
 5114  015702  005702                        TST     R2              ;SEE IF (!) WAS TYPED
 5115  015704  001731                        BEQ     10$             ;BR IF NOT (!)
 5116                          ;PROPAGATE CURRENT ITERATION NO. TO THE END OF TSTLST
 5117  015706  010102          16$:  MOV     R1,R2           ;COPY INDEX
 5118  015710  062702  005630                ADD     #TSTLST,R2      ;GET POSITION IN LIST
 5119  015714  022702  005642                CMP     #DFLTST-2,R2    ;SEE IF AT END OF LIST
 5120  015720  001002                        BNE     17$             ;BR IF NOT DONE
 5121  015722  000137  015214                JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
 5122  015726  016161  005630  005632  17$:  MOV     TSTLST(R1),TSTLST+2(R1) ;PROPAGATE TO NEXT WORD
 5123  015734  062701  000002                ADD     #2,R1           ;INCREMENT INDEX
 5124  015740  000762                        BR      16$             ;BR TO CONTINUE
 5125  015742  104401  005264          20$:  TYPE    ,BUFFO          ;ECHO BAD I  UT
 5126  015746  104401  001314                TYPE    ,$QUES          ;TYPE <?>  ' <CR>,<LF>
 5127  015752  000671                        BR      8$              ;GO ASK AG. .
 5128
 5129                          ;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
 5130  015754  104401  010173  INPUTP: TYPE    ,EXPLAN         ;EXPLAIN INPUT MODES
 5131  015760  005037  005502  ASKMDE: CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
 5132  015764  004737  031302          JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
 5133  015770  104401  010321          TYPE    ,PARMDE         ;ASK FOR DESIRED INPUT MODE
 5134  015774  104401  013274          TYPE    ,PROMPT         ;TYPE "* "
 5135  016000  004737  034002          JSR     PC,RDCHRS       ;READ RESPONSE TO MODE QUESTION
 5136  016004  014374                  DRVTST                  ;(^C) RETURN ADDRESS
 5137  016006  015760                  ASKMDE                  ;(^Z) RETURN ADDRESS
 5138  016010  015760                  ASKMDE                  ;(^U) OR ERROR RETURN ADDRESS
 5139  016012  005700                  TST     R0              ;SEE IF NULL INPUT
 5140  016014  001005                  BNE     4$              ;BR IF ANY INPUT
 5141  016016  104401  005264    2$:   TYPE    ,BUFFO          ;ECHO BAD INPUT
 5142  016022  104401  001314          TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
```

```
5143  016026  000754                          BR      ASKMDE          ;GO ASK AGAIN
5144  016030  022737  000124  005264  4$:     CMP     #'T,BUFFO       ;SEE IF (T) TYPED
5145  016036  001002                          BNE     6$              ;BR IF NOT (T)
5146  016040  000137  016172                  JMP     TYPLST          ;JUMP TO THE TYPE LIST ROUTINE
5147  016044  022737  000117  005264  6$:     CMP     #'O,BUFFO       ;SEE IF (O) TYPED
5148  016052  001002                          BNE     8$              ;BR IF NOT (O)
5149  016054  000137  016444                  JMP     OPNLST          ;JUMP TO THE OPEN LIST ROUTINE
5150  016060  022737  000123  005264  8$:     CMP     #'S,BUFFO       ;SEE IF (S) TYPED
5151  016066  001002                          BNE     10$             ;BR IF NOT (S)
5152  016070  000137  016730                  JMP     SETPRM          ;JUMP TO THE SET INDIV. PARAM. ROUTINE
5153  016074  022737  000122  005264  10$:    CMP     #'R,BUFFO       ;SEE IF (R) TYPED
5154  016102  001345                          BNE     2$              ;BR IF NOT (R), TO ECHO BAD INPUT
5155  016104  023737  005670  005672          CMP     S0,S1           ;COMPARE 20(DEC) SECTOR LIMITS
5156  016112  003403                          BLE     12$             ;BR IF S0 NOT > S1
5157  016114  104401  010427                  TYPE    ,SECNL1         ;TYPE "S0>S1 NOT ALLOWED"
5158  016120  000717                          BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
5159  016122  023737  005674  005676  12$:    CMP     S2,S3           ;COMPARE 22(DEC) SECTOR LIMITS
5160  016130  003405                          BLE     14$             ;BR IF S2 NOT > S3
5161  016132  104401  010457                  TYPE    ,SECNL2         ;TYPE "S2>S3"
5162  016136  104401  010440                  TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
5163  016142  000706                          BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
5164  016144  023737  005664  005666  14$:    CMP     FT,LT           ;COMPARE CHOSEN TRACK LIMITS
5165  016152  003405                          BLE     20$             ;BR IF FT NOT > LT
5166  016154  104401  010471                  TYPE    ,TRKNLW         ;TYPE "FT>LT"
5167  016160  104401  010440                  TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
5168  016164  000675                          BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
5169  016166  000137  017164          20$:    JMP     RUNTST          ;GO RUN THE TESTS
5170
5171
5172
5173                                  .SBTTL  TO - TYPE (T) LIST ROUTINE
5174                                  ;*THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
5175                                  ;*CONSOLE, LINE BY LINE. EACH LINE HAS THE FOLLOWING
5176                                  ;*FORMAT:  XX=YYYYYY , WHERE XX IS A PARAMETER MNEMONIC
5177                                  ;*AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
5178                                  ;*TYPING (^C) CAUSES IMMEDIATE RETU   TO DRVTST,
5179                                  ;*AND (^Z) CAUSES RETURN TO ASKMDE.
5180
5181  016172                          TYPLST:
5182  016172  005001                          CLR     R1              ;INITIALIZE INDEX
5183  016174  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5184  016200  004737  034350          1$:     JSR     PC,TYPPRM       ;TYPE CURRENT PARAMETER AND VALUE
5185  016204  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5186  016210  005737  005522                  TST     INTCHR          ;SEE IF ANY INPUT AT KBD
5187  016214  001420                          BEQ     8$              ;BR IF NO INPUT
5188  016216  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
5189  016224  001002                          BNE     4$              ;BR IF NOT (^C)
5190  016226  000137  014374                  JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
5191  016232  122737  000032  005522  4$:     CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
5192  016240  001002                          BNE     6$              ;BR IF NOT (^Z)
5193  016242  000137  015760                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5194  016246  004737  030556          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
5195  016252  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5196  016256  022761  040515  006010  8$:     CMP     #'MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
5197  016264  001002                          BNE     10$             ;BR IF NOT (MA)
5198  016266  062701  000002                  ADD     #2,R1           ;INCREMENT PARAMETER INDEX
```

```
5199  016272  062701  000002          10$:    ADD     #2,R1           ;INCREMENT PARAMETER INDEX
5200  016276  010102                          MOV     R1,R2           ;GET COPY OF INDEX
5201  016300  062702  005660                  ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
5202  016304  022702  005706                  CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
5203  016310  001333                          BNE     1$              ;BR IF NOT DONE YET
5204  016312  032737  100000  005700          BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
5205  016320  001005                          BNE     12$             ;BR IF PATTERN SPECIFIED
5206  016322  042777  000100  162614          BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
5207  016330  000137  015760                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5208                                  ;TYPE OUT USER-DEFINED PATTERN 15
5209  016334  104401  010616          12$:    TYPE    ,PFIFTN         ;TYPE 'USER-DEFINED PATTERN 15 :'
5210  016340  005001                          CLR     R1              ;INITIALIZE WORD INDEX
5211  016342  005737  005522          14$:    TST     INTCHR          ;SEE IF ANY INPUT
5212  016346  001420                          BEQ     20$             ;BR IF NO INPUT
5213  016350  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
5214  016356  001002                          BNE     16$             ;BR IF NOT (^C)
5215  016360  000137  014374                  JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
5216  016364  122737  000032  005522  16$:    CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
5217  016372  001002                          BNE     18$             ;BR IF NOT (^Z)
5218  016374  000137  015760                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5219  016400  004737  030556          18$:    JSR     PC,ECOBAD       ;ECHO BAD INPUT
5220  016404  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5221  016410  004737  034430          20$:    JSR     PC,TYPPAT       ;TYPE WORD XX = YYYYYY
5222  016414  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5223  016420  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
5224  016424  022701  000040                  CMP     #32.,R1         ;SEE IF 16 WORDS TYPED
5225  016430  001344                          BNE     14$             ;BR IF NOT DONE YET
5226  016432  042777  000100  162504          BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
5227  016440  000137  015760                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5228
5229
5230
5231                                  .SBTTL  TO - OPEN (O) LIST ROUTINE
5232                                  ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
5233                                  ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
5234                                  ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
5235                                  ;*TTY INPUT. TYPING (^C) CAUSES IMMEDIATE RETURN TO
5236                                  ;*DRVTST, AND (^Z) CAUSES RETURN TO ASKMDE.
5237
5238
5239  016444                          OPNLST:
5240  016444  104401  010534                  TYPE    ,DFQUES         ;ASK IF ALL DEFAULT VALUES DESIRED
5241  016450  004737  034002                  JSR     PC,RDCHRS       ;READ RESPONSE TO DEFAULT QUESTION
5242  016454  014374                          DRVTST                  ;(^C) RETURN ADDRESS
5243  016456  015760                          ASKMDE                  ;(^Z) RETURN ADDRESS
5244  016460  016444                          OPNLST                  ;(^U) OR ERROR RETURN ADDRESS
5245  016462  005700                          TST     R0              ;SEE IF NULL INPUT
5246  016464  001433                          BEQ     NOTDFT          ;BR IF DEFAULTS NOT REQUESTED
5247  016466  022737  000104  005264          CMP     #'D,BUFFO       ;SEE IF (D) TYPED
5248  016474  001405                          BEQ     LODFPT          ;BR IF DEFAULTS DESIRED
5249  016476  104401  005264                  TYPE    ,BUFFO          ;ECHO BAD INPUT
5250  016502  104401  001314                  TYPE    ,$QUES
5251  016506  000756                          BR      OPNLST          ;GO ASK AGAIN
5252                                  ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
5253  016510  012700  005706          LODFPT: MOV     #PRDFLT,R0      ;LOAD DEFAULT LIST ADDRESS
5254  016514  012702  005660                  MOV     #PRMLST,R2      ;LOAD PARAMETER LIST ADDRESS
```

```
5255  016520  012022                  4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
5256  016522  022702  005706                  CMP     #PRDFLT,R2      ;SEE IF DONE YET
5257  016526  001374                          BNE     4$              ;BR IF MORE VALUES TO LOAD YET
5258  016530  105737  003116                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE
5259  016534  001005                          BNE     6$              ;BR IF NOT DEFAULT MODE
5260  016536  012737  000001  023746          MOV     #1,$EOPCT       ;SET PASS COUNT = 1
5261  016544  000137  017246                  JMP     STPASS          ;GO RUN DFLT TESTS
5262  016550  000137  015760          6$:     JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5263  016554  005001                  NOTDFT: CLR     R1              ;INITIALIZE INDEX
5264                                  ;TYPE CURRENT PARAMETER AND VALUE
5265  016556  004737  034350          8$:     JSR     PC,TYPPRM       ;TYPE PARAMETER AND VALUE
5266  016562  104401  013253                  TYPE    ,SPACE1         ;TYPE A SPACE
5267  016566  104401  013274                  TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
5268                                  ;READ AND CHECK INPUT, IF ANY
5269  016572  004737  034002                  JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
5270  016576  014374                          DRVTST                  ;(^C) RETURN ADDRESS FOR RDCHRS
5271  016600  015760                          ASKMDE                  ;(^Z) RETURN ADDRESS FOR RDCHRS
5272  016602  016556                          8$                      ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
5273  016604  005700                          TST     R0              ;SEE IF ANY INPUT
5274  016606  001007                          BNE     10$             ;BR IF ANY INPUT
5275  016610  022761  040515  006010          CMP     #'MA,PRMNEM(R1) ;SEE IF (MA) JUST DEFAULTED
5276  016616  001023                          BNE     12$             ;BR IF NOT (MA)
5277  016620  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
5278  016624  000420                          BR      12$             ;BR TO MOVE ON TO NEXT PARAMETER
5279  016626  022700  000010          10$:    CMP     #10,R0          ;SEE IF 8 CHARACTERS TYPED
5280  016632  002431                          BLT     14$             ;BR IF MORE THAN 8 TYPED
5281  016634  012746  005264                  MOV     #BUFF0,-(SP)    ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
5282  016640  004737  054456                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
5283  016644  016716                          14$                     ;ERROR RETURN ADDRESS FOR OCTBIN
5284  016646  012637  005470                  MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
5285  016652  013737  054610  005472          MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
5286                                  ;CHECK PARAMETER VALUE AND PUT INTO LIST
5287  016660  004737  035006                  JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAM VALUE
5288  016664  016716                          14$                     ;ERROR RETURN ADDR. FOR CHKPRM
5289                                  ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5290  016666  004737  034462          12$:    JSR     PC,MODP15
5291                                  ;MOVE ON TO NEXT PARAMETER
5292  016672  062701  000002                  ADD     #2,R1           ;INCREMENT THE PARAMETER INDEX
5293  016676  010102                          MOV     R1,R2           ;GET COPY OF INDEX
5294  016700  062702  005660                  ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
5295  016704  022702  005706                  CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
5296  016710  001322                          BNE     8$              ;BR IF NOT DONE YET
5297  016712  000137  015760                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
5298  016716  104401  005264          14$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
5299  016722  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5300  016726  000713                          BR      8$              ;BR TO ASK AGAIN
5301
5302
5303
5304                                  .SBTTL  TO - SET (S) INDIVIDUAL PARAM ROUTINE
5305                                  ;*THIS ROUTINE IS USED TO ALLOW SETTING OF ANY INDIVIDUAL
5306                                  ;*PARAMETER VALUE BY TTY INPUT. A PROMPTER (>) IS
5307                                  ;*TYPED AND THE ROUTINE WAITS FOR INPUT OF THE FORM
5308                                  ;* XX=YYYYYY, WHERE XX IS THE MNEMONIC FOR ANY VALID
5309                                  ;*PARAMETER IN THE LIST, AND YYYYYY IS THE DESIRED PARAMETER
5310                                  ;*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.
                                        SETPRM:
016730  104401  013277                          TYPE    ,PRMPSP         ;TYPE '> ' PROMPTER
                                        ;READ AND CHECK INPUT, IF ANY
016734  004737  034002                          JSR     PC,RDCHRS       ;READ INPUT LINE
016740  014374                                  DRVTST                  ;(^C) RETURN ADDRESS FOR RDCHRS
016742  015760                                  ASKMDE                  ;(^Z) RETURN ADDRESS FOR RDCHRS
016744  016730                                  SETPRM                  ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
016746  020027  000004                          CMP     R0,#4           ;SEE IF AT LEAST 4 CHARACTERS TYPED
016752  002005                                  BGE     6$              ;BR IF AT LEAST 4 TYPED
016754  104401  005264                  4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
016760  104401  001314                          TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
016764  000761                                  BR      SETPRM          ;BR TO ASK FOR INPUT AGAIN
016766  020027  000013                  6$:     CMP     R0,#11.         ;SEE IF ELEVEN OR LESS CHARS TYPED
016772  003370                                  BGT     4$              ;BR IF MORE THAN ELEVEN TYPED
                                        ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
016774  005001                                  CLR     R1              ;INITIALIZE PARAMETER INDEX
016776  023761  005264  006010          8$:     CMP     BUFFO,PRMNEM(R1) ;TRY TO MATCH 2-CHAR MNEMONIC
017004  001411                                  BEQ     10$             ;BR IF PARAMETER FOUND IN TABLE
017006  062701  000002                          ADD     #2,R1           ;INCREMENT INDEX
017012  010102                                  MOV     R1,R2           ;GET COPY OF INDEX
017014  062702  005660                          ADD     #PRMLST,R2      ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
017020  022702  005706                          CMP     #PRDFLT,R2
017024  001364                                  BNE     8$              ;BR IF MORE TO CHECK YET
017026  000752                                  BR      4$              ;BR TO ECHO BAD INPUT
                                        ;CHECK FOR VALID LINE FORMAT
017030  012702  000002                  10$:    MOV     #2,R2           ;INITIALIZE BUFFER POINTER
017034  122762  000075  005264                  CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS "="
017042  001411                                  BEQ     12$             ;BR IF IT IS "="
017044  122762  000040  005264                  CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHAR IS A SPACE
017052  001340                                  BNE     4$              ;BR TO ECHO BAD INPUT
017054  005202                                  INC     R2              ;INCREMENT BUFFER POINTER
017056  122762  000075  005264                  CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS "="
017064  001333                                  BNE     4$              ;BR TO ECHO INVALID INPUT
017066  005202                          12$:    INC     R2              ;INCREMENT BUFFER POINTER
017070  020200                                  CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
017072  002330                                  BGE     4$              ;BR TO ECHO INVALID INPUT
017074  122762  000040  005264                  CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHARACTER IS SPACE
017102  001003                                  BNE     14$             ;BR IF NOT A SPACE
017104  005202                                  INC     R2              ;INCREMENT BUFFER POINTER
017106  020200                                  CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
017110  002321                                  BGE     4$              ;BR IF NONE LEFT
017112  160200                          14$:    SUB     R2,R0           ;SUBTRACT POINTER FROM CHAR COUNT
017114  020027  000010                          CMP     R0,#10          ;SEE HOW MANY CHARS LEFT
017120  003315                                  BGT     4$              ;BR IF TOO MANY LEFT
                                        ;CHECK FOR LEGAL PARAMETER VALUE
017122  062702  005264                          ADD     #BUFFO,R2       ;GET ADDRESS OF DIGITS
017126  010246                                  MOV     R2,-(SP)        ;PUT IT ON STACK FOR OCTBIN
017130  004737  054456                          JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
017134  016754                                  4$                      ;ERROR RETURN ADDR. FOR OCTBIN
017136  012637  005470                          MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
017142  013737  054610  005472                  MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
017150  004737  035006                          JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAMETER VALUE
```

```
5367  017154  016754                        4$                      ;ERROR RETURN ADDR. FOR CHKPRM
5368                                  ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5369  017156  004737  034462                JSR     PC,MODP15
5370  017162  000662                        BR      SETPRM          ;RETURN TO ASK FOR ANOTHER PARAMETER
5371
5372
5373
5374                                  .SBTTL  TO - RUN (R) TESTS ROUTINE
5375                                  ;*THIS ROUTINE IS ENTERED TO SET UP THE RUNNING OF TESTS,
5376                                  ;*WHOSE PARAMETERS HAVE BEEN DETERMINED. IT FIRST ASKS
5377                                  ;*FOR THE DESIRED NUMBER OF PROGRAM PASSES, AND STORES
5378                                  ;*THIS IN $EOPCT FOR THE END OF PASS ROUTINE. THEN, THE
5379                                  ;*DRIVE LIST IS SCANNED FOR THE FIRST DESIRED DRIVE, AND
5380                                  ;*THE HEADER ON SECTOR 0 IS READ TO GET THE FORMAT FOR THIS
5381                                  ;*DRIVE. THIS IS STORED IN THE FORMAT BYTE, AND A JUMP IS
5382                                  ;*MADE TO THE FIRST TEST.
5383                                  ;*THE END OF PASS ROUTINE RETURNS TO 'NEWDRV" TO SELECT
5384                                  ;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
5385                                  ;* (ALL DRIVES) IS COMPLETED.
5386
5387  017164                          RUNTST:
5388                                  ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
5389  017164  104401  010742          4$:     TYPE    ,ENTPAS         ;ASK FOR NUMBER OF PASSES
5390  017170  004737  034002                  JSR     PC,RDCHRS       ;READ RESPONSE
5391  017174  014374                        DRVTST                  ;(^C) RETURN ADDRESS
5392  017176  015760                        ASKMDE                  ;(^Z) RETURN ADDRESS
5393  017200  017164                        4$                      ;(^U) OR ERROR RETURN ADDRESS
5394  017202  005700                        TST     R0              ;SEE IF NULL INPUT
5395  017204  001403                        BEQ     6$              ;GO ASK AGAIN
5396  017206  022700  000005                CMP     #5,R0           ;SEE HOW MANY CHARS TYPED
5397  017212  002005                        BGE     8$              ;BR IF 5 OR LESS
5398  017214  104401  005264          6$:     TYPE    ,BUFF0          ;ECHO BAD INPUT
5399  017220  104401  001314                  TYPE    ,$QUES
5400  017224  000757                        BR      4$              ;GO ASK AGAIN
5401  017226  012746  005264          8$:     MOV     #BUFF0,-(SP)    ;GET BUF ADDR ON STACK FOR OCTBIN
5402  017232  004737  054456                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
5403  017236  017214                        6$                      ;ERROR RETURN ADDRESS FOR OCTBIN
5404  017240  012637  023746                  MOV     (SP)+,$EOPCT    ;SET DESIRED NO. OF PASSES (1-77777)
5405  017244  001763                        BEQ     6$              ;BR IF 0, TO ASK AGAIN
5406                                  ;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
5407  017246  005037  001326          STPASS: CLR     $PASS           ;INIT. THE PASS NUMBER
5408  017252  112737  000001  003120          MOVB    #1,TSTING       ;SET "RUNNING TESTS" FLAG
5409  017260  012737  177777  005500  NEWPAS: MOV     #177777,DRIVE   ;INITIALIZE DRIVE NO. TO -1
5410  017266  005037  001330                  CLR     $DEVCT          ;INIT APT DEVICE COUNT TO 0
5411                                  ;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
5412  017272  012737  000000  177776  NEWDRV: MOV     #PR0,@#PS       ;RE-ESTABLISH PRIORITY 0
5413  017300  012706  001100                  MOV     #STACK,SP       ;RESTORE THE STACK
5414  017304  105037  003130                  CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
5415  017310  005037  005522                  CLR     INTCHR          ;CLEAR TTY INPUT BUFFER WORD
5416  017314  005237  005500                  INC     DRIVE           ;INCREMENT DRIVE NUMBER
5417  017320  022737  000010  005500          CMP     #10,DRIVE       ;SEE IF DONE WITH THIS PASS
5418  017326  001002                        BNE     2$              ;BR IF NOT DONE CHECKING LIST
5419  017330  000137  023714                  JMP     DUNPAS          ;JUMP IF DONE WITH THIS PASS
5420  017334  013700  005500          2$:     MOV     DRIVE,R0        ;GET NEW DRIVE NUMBER
5421  017340  105760  005610                  TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
5422  017344  001752                        BEQ     NEWDRV          ;BR IF NOT MARKED, TO CHECK ANOTHER
```

```
 5423  017346  105037  001102                 CLRB    $TSTNM          ;INIT TEST NUMBER TO 0
 5424  017352  005037  001304                 CLR     $TIMES          ;INIT ITERATION COUNT
 5425  017356  105760  005620         TYPED:  TSTB    DTYLST(R0)      ;SEE IF RK07
 5426  017362  001036                 TYPED:  BNE     1$              ;BR IF YES
 5427  017364  105037  003124                 CLRB    TYPFMT          ;ELSE LOAD ALL RK06 PARAM
 5428  017370  012737  000624  017776         MOV     #624,LCM6
 5429  017376  012737  000631  005736         MOV     #631,PRMLIM+2
 5430  017404  012737  000632  005662         MOV     #632,LC
 5431  017412  012737  000632  020000         MOV     #632,LSTCYL
 5432  017420  012737  000632  005710         MOV     #632,PRDFLT+2
 5433  017426  012737  000632  005742         MOV     #632,PRMLIM+6
 5434  017434  012737  000400  020002         MOV     #400,HOLD1
 5435  017442  012737  000025  020004         MOV     #25,HOLD3
 5436  017450  012737  010025  020006         MOV     #10025,HOLD4
 5437  017456  000436                         BR      3$
 5438
 5439  017460  152737  000004  003124  1$:    BISB    #B.CDT,TYPFMT   ;LOAD RK07 PARAMETERS
 5440  017466  012737  001450  017776         MOV     #1450,LCM6
 5441  017474  012737  001455  005736         MOV     #1455,PRMLIM+2
 5442  017502  012737  001456  005662         MOV     #1456,LC
 5443  017510  012737  001456  020000         MOV     #1456,LSTCYL
 5444  017516  012737  001456  005710         MOV     #1456,PRDFLT+2
 5445  017524  012737  001456  005742         MOV     #1456,PRMLIM+6
 5446  017532  012737  001000  020002         MOV     #1000,HOLD1
 5447  017540  012737  002025  020004         MOV     #2025,HOLD3
 5448  017546  012737  012025  020006         MOV     #12025,HOLD4
 5449
 5450  017554                         3$:
 5451                                 ;       TSTB    HDALI           ;ARE WE DOING JUST HEAD-ALIGNMENT?
 5452                                 ;       BEQ     13$             ;YES, SKIP NEXT
 5453                                 ;       TSTB    HDALI+1         ;WHICH RETURN
 5454                                 ;       BMI 29$                 ;SECOND RETURN
 5455                                 ;       JMP     @#INTGO         ;GO TO ALIGNMENT CODE
 5456                                 ;29$:   JMP     @#INTGON        ;GO TO AUTO ALIGNMENT CODE
 5457  017554  010037  001332         13$:    MOV     R0,$UNIT        ;SET DRV NO FOR APT
 5458                                 ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
 5459  017560  004737  032250                 JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
 5460  017564  112765  000125  000001         MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
 5461  017572  004737  042510                 JSR     PC,DRVCAL       ;DO READ HEADER
 5462  017576  013712  020004                 MOV     HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT = 0
 5463  017602  032712  000200         4$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
 5464  017606  001775                         BEQ     4$              ;BR IF NOT RDY YET
 5465  017610  013712  020006                 MOV     HOLD4,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
 5466  017614  032712  000200         6$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
 5467  017620  001775                         BEQ     6$              ;BR IF NOT RDY YET
 5468  017622  105037  003125                 CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
 5469  017626  013737  005674  005506         MOV     S2,FS           ;INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
 5470  017634  013737  005676  005510         MOV     S3,LS
 5471  017642  005762  000024                 TST     RKDB(R2)        ;POP SILO ONE TIME
 5472  017646  032762  001000  000024         BIT     #BIT9,RKDB(R2)  ;TEST FORMAT BIT IN HEADER WORD 2
 5473  017654  001411                         BEQ     8$              ;BR IF 22 SECTOR FORMAT
 5474  017656  152737  000020  003125         BISB    #B.CFMT,FORMAT  ;SET 20 SECTOR FORMAT FOR THIS DRIVE
 5475  017664  013737  005670  005506         MOV     S0,FS           ;INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
 5476  017672  013737  005672  005510         MOV     S1,LS
 5477  017700  013737  005704  005502  8$:     MOV     ST,STALLS       ;SET NUMBER OF UNIT STALLS DESIRED
 5478  017706  004737  032250                 JSR     PC,INITSS       ;INIT THE S.S.
```

```
 5479  017712  004737  041066                    JSR     PC,REDBSF       ;READ BAD SECTOR FILE FOR THIS DRIVE
 5480  017716  113737  005500  011070            MOVB    DRIVE,DRVNO     ;GET DRIVE NO.
 5481  017724  152737  000060  011070            BISB    #'0,DRVNO       ;CONVERT TO ASCII
 5482  017732  104401  011051                    TYPE    ,TSTDRN         ;TYPE "TESTING DRIVE X"
 5483  017736  005737  001326                    TST     $PASS           ;SEE IF THIS IS FIRST PASS
 5484  017742  001012                            BNE     10$             ;BR IF NOT FIRST PASS
 5485  017744  004737  035312                    JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
 5486  017750  004737  035432                    JSR     PC,CRTSER       ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
 5487  017754  032737  000020  005702            BIT     #BIT4,CS        ;SEE IF BAD SECTORS SHOULD BE TYPED
 5488  017762  001402                            BEQ     10$             ;BR IF NOT
 5489  017764  004737  041356                    JSR     PC,TYPBSF       ;TYPE BAD SECTOR FILES
 5490  017770  005037  005522            10$:    CLR     INTCHR          ;INIT. TTY INPUT CHAR BUFFER
 5491  017774  000405                            BR      TST1            ;;GO TO FIRST TEST
 5492
 5493  017776  000000                    LCM6:   0
 5494  020000  000000                    LSTCYL: 0
 5495  020002  000000                    HOLD1:  0
 5496  020004  000000                    HOLD3:  0
 5497  020006  000000                    HOLD4:  0
 5498
 5499
 5500
 5501
```

```
5502                                    ;;*****************************************************************
5503                                    ;*TEST 1        OFFSET-TO-FAILURE MEASUREMENTS
5504                                    ;*IN THIS TEST A REPEATED PATTERN 072307(OCT) IS WRITTEN INTO ALL
5505                                    ;*WORDS OF SECTORS FS AND LS ON TRACKS 0-2 OF CYLINDERS FC AND LC.
5506                                    ;*THE SAME SECTORS ON ADJACENT CYLINDERS ARE THEN WRITTEN WITH
5507                                    ;* 167230(OCT) (TO ESTABLISH A KNOWN PATTERN) , BUT THEY ARE NOT TESTED.
5508                                    ;*THEN, EACH DESIRED SECTOR IS WRITE CHECKED WITH INCREASING POSITIVE
5509                                    ;*OFFSETS, UNTIL EITHER A "DATA TYPE" ERROR OCCURS, OR THE FULL RANGE
5510                                    ;*OF POSITIVE OFFSETS HAVE BEEN APPLIED. THEN, THE SAME SECTOR IS WRITE
5511                                    ;*CHECKED WITH INCREASING NEGATIVE HEAD OFFSETS, UNTIL FAILURE OR
5512                                    ;*MAXIMUM NEGATIVE OFFSET IS APPLIED. THIS IS DONE FOR ALL THE SECTORS
5513                                    ;*ORIGINALLY WRITTEN, IN THE FOLLOWING ORDER : SECTORS FS AND LS ON
5514                                    ;*CYLINDERS FC AND LC FOR TRACK 0, THEN FOR TRACK 1, AND FINALLY FOR
5515                                    ;*TRACK 2. THE OFFSET-TO-FAILURE RESULTS ARE REPORTED IN ORDER OF
5516                                    ;*INCREASING TRACKS, AS FOLLOWS :
5517                                    ;*
5518                                    ;* OFFSET-TO-FAILURE MEASUREMENTS :
5519                                    ;*
5520                                    ;* TRACK  CYLN  SECT  +OFST  -OFST
5521                                    ;*                   (OCT)  (OCT)
5522                                    ;*   X    XXX    XX   XXXX   XXXX
5523                                    ;*   X    XXX    XX   XXXX   XXXX
5524                                    ;*   X    XXX    XX   XXXX   XXXX
5525                                    ;*         ETC.
5526                                    ;*
5527                                    ;*NOTE: IN AN ADRS 200 DEFAULT RUN, FS DEFAULTS TO 0, AND LS DEFAULTS
5528                                    ;*TO 10(DEC), FOR THIS TEST.
5529                                    ;*****************************************************************
5530  020010  000004                    TST1:   SCOPE
5531  020012  012737  000001  001324            MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5532  020020  004737  033640                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5533  020024  004737  033706                    JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5534  020030  000137  021136                    JMP     TST2            ;JUMP TO NEXT TEST
5535                                    ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5536  020034  004737  032250                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5537  020040  004737  030536                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5538                                    ;INITIALIZE PARAMETERS, WRITE THE SECTORS, AND WRITE CHECK THEM
5539  020044  105737  003116                    TSTB    MDFLAG          ;SEE IF ADRS 200 DEFAULT RUN
5540  020050  001005                            BNE     3$              ;BR IF NOT
5541  020052  005037  005506                    CLR     FS              ;SET FS = 0
5542  020056  012737  000012  005510            MOV     #10.,LS         ;SET LS = 10(DEC)
5543  020064  013765  005660  000002  3$:       MOV     FC,P.CYLN(R5)   ;SET CYL = FC
5544  020072  113765  005506  000004            MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS
5545  020100  105065  000005                    CLRB    P.TRCK(R5)      ;SET TRACK = 0
5546  020104  013737  005662  005504            MOV     LC,CYLNDR
5547  020112  023737  020000  005662            CMP     LSTCYL,LC       ;SEE IF LC =632/1456
5548  020120  001002                            BNE     4$              ;BR IF NOT 632/1456
5549  020122  005337  005504                    DEC     CYLNDR          ;MAKE IT 631 (PRESERVE BSF)
5550  020126  012703  072307          4$:       MOV     #72307,R3       ;GET DATA PATTERN
5551  020132  004737  037562                    JSR     PC,WRTSEC       ;DO WRITE AND WRITE CHECK
5552  020136  012703  167230                    MOV     #167230,R3      ;DATA FOR AJACENT CYLS
5553  020142  005365  000002                    DEC     P.CYLN(R5)      ;NEXT LOWER CYL
5554  020146  100402                            BMI     6$              ;BR IF CYL NEGATIVE
5555  020150  004737  037562                    JSR     PC,WRTSEC       ;DO WRITE AND WRITE CHECK
5556  020154  062765  000002  000002  6$:       ADD     #2,P.CYLN(R5)   ;NEXT HIGHER CYL
5557  020162  023765  020000  000002            CMP     LSTCYL,P.CYLN(R5)       ;SEE IF CYL = 632/1456
```

```
5558  020170  001004                          BNE     8$                ;BR IF NOT
5559  020172  122765  000002  000005          CMPB    #2,P.TRCK(R5)     ;SEE IF BSF
5560  020200  001402                          BEQ     10$               ;BR IF YES, TO PROTECT IT
5561  020202  004737  037562          8$:     JSR     PC,WRTSEC         ;DO WRITE AND WRITE CHECK
5562  020206  005365  000002          10$:    DEC     P.CYLN(R5)        ;RESTORE CYL NO.
5563  020212  105265  000005                  INCB    P.TRCK(R5)        ;INCR TRACK
5564  020216  122765  000003  000005          CMPB    #3,P.TRCK(R5)     ;TRACK = 3 YET ?
5565  020224  001340                          BNE     4$                ;BR IF NOT YET
5566  020226  126537  000004  005510          CMPB    P.SECT(R5),LS     ;SEE IF SECTOR = LS
5567  020234  001406                          BEQ     14$               ;BR IF YES
5568  020236  113765  005510  000004          MOVB    LS,P.SECT(R5)     ;SET SECTOR = LS
5569  020244  105065  000005          12$:    CLRB    P.TRCK(R5)        ;SET TRACK = 0
5570  020250  000726                          BR      4$                ;BR TO CONTINUE WRITING
5571  020252  026537  000002  005504  14$:    CMP     P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
5572  020260  001407                          BEQ     16$               ;BR IF YES
5573  020262  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET CYL = LC
5574  020270  113765  005506  000004          MOVB    FS,P.SECT(R5)     ;SET SECTOR = FS AGAIN
5575  020276  000762                          BR      12$               ;BR TO CONTINUE WRITING
5576                          ;INITIALIZE PARAMETERS FOR OFFSET MEASUREMENTS
5577  020300  032737  000002  005702  16$:    BIT     #BIT1,CS          ;SEE IF REPORTS ARE INHIBITED
5578  020306  001002                          BNE     18$               ;BR IF INHIBITED
5579  020310  104401  011177                  TYPE    ,OFSHED           ;TYPE HEADINGS
5580  020314  113765  005664  000005  18$:    MOVB    FT,P.TRCK(R5)     ;SET TRACK = FT
5581  020322  013765  005660  000002          MOV     FC,P.CYLN(R5)     ;SET CYLINDER = FC
5582  020330  113765  005506  000004          MOVB    FS,P.SECT(R5)     ;SET SECTOR = FS
5583                          ;LOAD THE R/W BUFFER FOR WRITE CHECKS
5584  020336  012700  065636                  MOV     #RWBUF,R0         ;SET BUFFER ADDRESS
5585  020342  012701  000400                  MOV     #400,R1           ;PREPARE TO LOAD 400(OCT) WORDS
5586  020346  012720  072307          19$:    MOV     #072307,(R0)+     ;LOAD A WORD INTO BUFFER
5587  020352  005301                          DEC     R1                ;DECR COUNTER
5588  020354  001374                          BNE     19$               ;BR IF NOT DONE YET
5589  020356  005004                  20$:    CLR     R4                ;INIT (+) OFFSET VALUE
5590  020360  005000                          CLR     R0                ;INIT OFFSET NUMBER TO 0
5591  020362  012737  177777  005546          MOV     #-1,PLOFST        ;INITIALIZE CURRENT FAILING (+) OFFSET
5592  020370  012737  177777  005550          MOV     #-1,NGOFST        ;INITIALIZE CURRENT FAILING (-) OFFSET
5593                          ;PERFORM OFFSETS AND WRITE CHECKS
5594  020376  112765  000117  000001  22$:    MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5595  020404  004737  042510                  JSR     PC,DRVCAL         ;PERFORM SEEK
5596  020410  110065  000006                  MOVB    R0,P.OFST(R5)     ;SET CURRENT OFFSET
55^7  020414  112765  000115  000001          MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET COMMAND
5598  020422  004737  042510                  JSR     PC,DRVCAL         ;PERFORM OFFSET
5599  020426  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
5600  020434  012737  021102  003046          MOV     #WRCKHD,A.ABNL    ;SET SPECIAL ERROR HANDLER ADDRESS
5601  020442  105037  003140                  CLRB    WCEFLG            ;CLEAR WRITE CHECK ERROR FLAG
5602  020446  004737  042510                  JSR     PC,DRVCAL         ;PERFORM A WRITE CHECK
5603  020452  012737  044156  003046          MOV     #ERRHDL,A.ABNL    ;RESTORE ERROR HANDLER ADDRESS
5604  020460  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYS CLEAR CMND
5605  020466  004737  042510                  JSR     PC,DRVCAL         ;CLEAR THE SUBSYSTEM
5606  020472  112765  000115  000001          MOVB    #OFFSET,P.CMND(R5) ;SET OFFSET CMND
5607  020500  105065  000006                  CLRB    P.OFST(R5)        ;SET OFFSET = 0
5608  020504  004737  042510                  JSR     PC,DRVCAL         ;OFFSET BACK TO 0
5609  020510  105737  003140                  TSTB    WCEFLG            ;SEE IF A WRITE CHECK ERROR OCCURRED
5610  020514  001020                          BNE     26$               ;BR IF AN ERROR OCCURRED
5611  020516  010001                          MOV     R0,R1             ;GET A COPY OF CURRENT OFFSET
5612  020520  005200                          INC     R0                ;INCREMENT OFFSET
5613  020522  062704  000031                  ADD     #25.,R4           ,INCR OFFSET VALUE BY 25(DEC) UIN
```

```
5614  020526  042701  177700                    BIC     #177700,R1      ;MASK FOR MAGNITUDE BITS
5615  020532  022701  000060                    CMP     #60,R1          ;SEE IF MAX OFFSET APPLIED IN THIS DIRECTION
5616  020536  001317                            BNE     22$             ;BR IF NOT YET
5617  020540  032700  000200                    BIT     #BIT7,R0        ;SEE IF NEG OFFSETS DONE YET
5618  020544  001015                            BNE     32$             ;BR IF YES, TO PRINT CURRENT SECTOR RESULT
5619  020546  012700  000200          24$:      MOV     #200,R0         ;INIT FOR NEG OFFSETS
5620  020552  005004                            CLR     R4              ;INIT (-) OFFSET VALUE TO 0
5621  020554  000710                            BR      22$             ;BR TO APPLY NEG OFFSETS
5622  020556  032700  000200          26$:      BIT     #BIT7,R0        ;SEE IF NEG OFFSETS DONE YET
5623  020562  001403                            BEQ     28$             ;BR IF NOT YET
5624  020564  010437  005550                    MOV     R4,NGOFST       ;SAVE THIS FAILING (-) OFFSET VALUE
5625  020570  000403                            BR      32$             ;BR TO PRINT CURRENT SECTOR RESULT
5626  020572  010437  005546          28$:      MOV     R4,PLOFST       ;SAVE THIS FAILING (+) OFFSET VALUE
5627  020576  000763                            BR      24$             ;BR TO APPLY NEGATIVE OFFSETS
5628  020600  032737  000002  005702  32$:      BIT     #BIT1,CS        ;SEE IF REPORTS INHIBITED
5629  020606  001051                            BNE     43$             ;BR IF INHIBITED
5630                                  ;TYPE OFFSET RESULTS FOR THIS SECTOR
5631  020610  116501  000005                    MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
5632  020614  010146                            MOV     R1,-(SP)        ;PUT IT ON STACK
5633  020616  104403                            TYPOS                   ;TYPE IT
5634  020620     003                            .BYTE   3               ;3 DIGITS
5635  020621     000                            .BYTE   0               ;SUPPRESS
5636  020622  104401  013251                    TYPE    ,SPACE3
5637  020626  016546  000002                    MOV     P.CYLN(R5),-(SP) ;PUT CYL ON STACK
5638  020632  104403                            TYPOS                   ;TYPE IT
5639  020634     004                            .BYTE   4               ;4 DIGITS
5640  020635     000                            .BYTE   0               ;SUPPRESS
5641  020636  104401  013252                    TYPE    ,SPACE2
5642  020642  116501  000004                    MOVB    P.SECT(R5),R1   ;GET SECTOR
5643  020646  010146                            MOV     R1,-(SP)        ;PUT IT ON STACK
5644  020650  104404                            TYPON                   ;TYPE SECTOR
5645  020652  104401  013253                    TYPE    ,SPACE1
5646  020656  005737  005546                    TST     PLOFST          ;SEE IF THERE WAS A FAILING (+) OFFSET
5647  020662  100003                            BPL     36$             ;BR IF THERE WAS
5648  020664  104401  011170                    TYPE    ,NOFALS         ;TYPE "  NONE"
5649  020670  000403                            BR      38$             ;CONTINUE
5650  020672  013746  005546          36$:      MOV     PLOFST,-(SP)    ;PUT (+) OFFSET VALUE ON STACK
5651  020676  104405                            TYPDS                   ;TYPE IT IN DECIMAL
5652  020700  104401  013253          38$:      TYPE    ,SPACE1
5653  020704  005737  005550                    TST     NGOFST          ;SEE IF THERE WAS A FAILING (-) OFFSET
5654  020710  100003                            BPL     40$             ;BR IF THERE W'S
5655  020712  104401  011170                    TYPE    ,NOFALS         ;TYPE "  NONE"
5656  020716  000403                            BR      42$             ;CONTINUE
5657  020720  013746  005550          40$:      MOV     NGOFST,-(SP)    ;PUT (-) OFFSET VALUE ON STACK
5658  020724  104405                            TYPDS                   ;TYPE IT IN DECIMAL
5659  020726  104401  001315          42$:      TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5660                                  ;SET UP TO TEST NEXT SECTOR
5661  020732  126537  000004  005510  43$:      CMPB    P.SECT(R5),LS   ;SEE IF SECTOR = LS
5662  020740  001404                            BEQ     44$             ;BR IF YES
5663  020742  116565  005510  000004            MOVB    LS,P.SECT(R5)   ;SET SECTOR = LS
5664  020750  000602                            BR      20$             ;GO TEST THIS SECTOR
5665  020752  026537  000002  005504  44$:      CMP     P.CYLN(R5),CYLNDR ;SEE IF CYL = LC
5666  020760  001410                            BEQ     48$             ;BR IF YES
5667  020762  013765  005504  000002            MOV     CYLNDR,P.CYLN(R5) ;SET CYL = LC
5668  020770  113765  005506  000004  46$:      MOVB    FS,P.SECT(R5)   ;SET SECTOR = FS
5669  020776  000137  020356                    JMP     20$             ;GO TEST THIS SECTOR
```

```
 5670  021002  105265  000005          48$:    INCB    P.TRCK(R5)        ;INCREMENT TRACK
 5671  021006  122765  000003  000005          CMPB    #3,P.TRCK(R5)     ;SEE IF TRACK = 3
 5672  021014  001404                          BEQ     50$               ;BR IF YES
 5673  021016  013765  005660  000002          MOV     FC,P.CYLN(R5)     ;SET CYL = FC
 5674  021024  000761                          BR      46$               ;GO TEST THIS SECTOR
 5675  021026  112765  000113  000001  50$:    MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
 5676  021034  004737  042510                  JSR     PC,DRVCAL         ;DO CLEANUP RECALIBRATE
 5677  021040  013737  005674  005506          MOV     S2,FS             ;RESTORE FS,LS FOR 22 SECTORS
 5678  021046  013737  005676  005510          MOV     S3,LS
 5679  021054  105737  003125                  TSTB    FORMAT            ;SEE IF 22 SECTORS
 5680  021060  001406                          BEQ     54$               ;BR IF YES
 5681  021062  013737  005670  005506          MOV     S0,FS             ;RESTORE FS,LS FOR 20 SECTORS
 5682  021070  013737  005672  005510          MOV     S1,LS
 5683  021076                          54$:
 5684  021076  000137  021136                  JMP     TST2              ;JUMP TO NEXT TEST
 5685
 5686                                  ;*THIS IS THE ABNORMAL RETURN FROM THE DRIVER, USED WHEN A
 5687                                  ;*   "DATA TYPE" ERROR IS EXPECTED (AND VALID).
 5688  021102  032765  040000  000020  WRCKHD: BIT     #WCE,P.CS2(R5)    ;SEE IF WRITE CHECK ERROR OCCURRED
 5689  021110  001006                          BNE     4$                ;BR IF WCE
 5690                                  ;CHECK FOR OTHER "DATA TYPE" ERROR
 5691  021112  032765  130400  000034          BIT     #HVRC!DTE!OPI!DCK,P.ER(R5)
 5692  021120  001002                          BNE     4$                ;BR IF A DATA ERROR OCCURRED
 5693  021122  000137  044156                  JMP     ERRHDL            ;GO HANDLE OTHER ERROR
 5694  021126  105237  003140          4$:     INCB    WCEFLG            ;SET WRITE CHECK ERROR FLAG
 5695  021132  000137  046334                  JMP     RETNML            ;TAKE NORMAL RETURN
 5696
 5697
 5698
 5699                                  ;;*****************************************************************
 5700                                  ;*TEST 2        NPR/MEMORY WORD ADDRESSING TEST
 5701                                  ;*IN THIS TEST, MEMORY WORD ADDRESSING CAPABILITY DURING RK06 NPR'S,
 5702                                  ;*IS TESTED.
 5703                                  ;*BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
 5704                                  ;*THEN, STARTING AT THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK
 5705                                  ;*BEYOND ADDRESS RWBUF+6000(OCTAL) , WRITE UNIQUE NUMBERS
 5706                                  ;*(STARTING WITH 1), INTO EACH OF UP TO 64K WORDS (DEPENDING UPON THE
 5707                                  ;*AMOUNT OF PHYSICAL MEMORY). THESE NUMBERS REFLECT UNIQUE ADDRESSES
 5708                                  ;*WITHIN THE 64K BLOCK. NEXT, WRITE THE 64K WORDS (MAX) ONTO THE DISK
 5709                                  ;*AT ADDRESS FC,FT,FS (SCALED, IF NECESSARY, TO AVOID PACK OVERFLOW
 5710                                  ;*OR DESTRUCTION OF THE BAD SECTOR FILE). THEN ZERO THE ENTIRE BLOCK
 5711                                  ;*IN MEMORY, AND READ THE DATA BACK FROM THE DISK, AT THE SAME PHYSICAL
 5712                                  ;*ADDRESSES. COMPARE THE DATA READ TO THE DATA WRITTEN. TYPE THE FAILING
 5713                                  ;*VIRTUAL (IF MEM. MGT.) AS WELL AS PHYSICAL ADDRESSES, AND THE GOOD
 5714                                  ;*AND BAD DATA, FOR UP TO THE FIRST 10(DEC) FAILING LOCATIONS.
 5715                                  ;* IF MEMORY MANAGEMENT IS INSTALLED AND THERE IS ADDITIONAL MEMORY TO
 5716                                  ;*EXERCISE, REPEAT THE ABOVE WORD ADDRESSING TEST, FOR EACH OF THE
 5717                                  ;*REMAINING 64K MEMORY BLOCKS.
 5718                                  ;;*****************************************************************
 5719  021136  000004                  TST2:   SCOPE
 5720  021140  012737  000002  001324          MOV     #2,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
 5721  021146  004737  033640                  JSR     PC,SETTUP         ;SET UP FOR LOOP ON ERROR
 5722  021152  004737  033706                  JSR     PC,CHKITR         ;SEE IF ITER. NO. = 0 FOR THIS TEST
 5723  021156  000137  021446                  JMP     TST3              ;JUMP TO NEXT TEST
 5724                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
 5725  021162  004737  032250                  JSR     PC,INITSS         ;INITIALIZE DRIVER PARAMS AND SUB-SYS
```

```
5726  021166  004737  030536              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5727                            ;SAVE XXDP LOADER, IF PRESENT
5728  021172  004737  031730              JSR     PC,FNDXDP       ;COMPUTE STARTING ADR OF XXDP
5729  021176  012737  065636  005542      MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADR
5730  021204  005037  005544              CLR     XDPSAV+2
5731  021210  004737  032002              JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
5732  021214  112737  000001  003133      MOVB    #1,XOVLAD       ;SET INDICATORS
5733  021222  112737  000001  003134      MOVB    #1,XDPSVD
5734  021230  012737  177777  005536      MOV     #-1,PATRN       ;SET FLAG FOR SVPRMS
5735                            ;INIT PARAMETERS, BEGIN TESTING
5736  021236  004737  040612              JSR     PC,SETUPM       ;INIT PARAMS
5737  021242  004737  035236      4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WORD COUNT FOR THIS MA
5738  021246  005701                      TST     R1              ;SEE IF ALL MEMORY TESTED YET
5739  021250  100467                      BMI     50$             ;BR IF ALL DONE
5740  021252  005065  000012              CLR     P.WC(R5)        ;INIT WORD COUNT TO 65,536(DEC)
5741  021256  005701                      TST     R1              ;SEE IF WRD CNT SHOULD BE 65,536
5742  021260  001003                      BNE     6$              ;BR IF YES
5743  021262  005400                      NEG     R0
5744  021264  010065  000012              MOV     R0,P.WC(R5)     ;SET WORD COUNT
5745  021270  013765  005600  000010  6$: MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
5746  021276  013701  005602              MOV     MA+2,R1         ;GET MA BITS 16-21
5747  021302  042701  177774              BIC     #177774,R1      ;MASK FOR LO 2 BITS
5748  021306  150165  000007              BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
5749  021312  005737  057710              TST     $KT11           ;SEE IF MEM MGT PRESENT
5750  021316  100010                      BPL     14$             ;BR IF NOT
5751                            ;SET UP MEM MGT
5752  021320  013737  005600  005604      MOV     MA,PMA          ;SET BUFFER ADDRESS
5753  021326  013737  005602  005606      MOV     MA+2,PMA+2
5754  021334  004737  031606              JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5755                            ;PERFORM THE TESTS
5756  021340  012737  021340  001110  14$: MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
5757  021346  004737  033640              JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5758  021352  004737  042020              JSR     PC,SVPRMS       ;SAVE PARAMS FOR THIS XFER
5759  021356  004737  037656              JSR     PC,LDMEM1       ;LOAD THIS MEM BLK WITH INCREMENTING DATA
5760  021362  112765  000123  000001      MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5761  021370  004737  042132              JSR     PC,TRNSFR       ;WRITE THE DATA
5762  021374  005003                      CLR     R3              ;SET DATA WORD = 0
5763  021376  004737  037670              JSR     PC,LDMEM2       ;LOAD MEM BLK WITH ALL ZEROS
5764  021402  112765  000121  000001      MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
5765  021410  004737  042132              JSR     PC,TRNSFR       ;READ THE DATA FROM DISK
5766  021414  004737  040022              JSR     PC,CKMEM1       ;PERFORM SOFTWARE COMPARE OF DATA
5767  021420  104411                      SCOPER                  ;CHECK FOR INTERNAL LOOP ON ERROR
5768                            ;GET NEW MA FOR NEXT TRANSFER
5769  021422  004737  041020              JSR     PC,INCRMA       ;COMPUTE NEXT MA
5770  021426  000705                      BR      4$              ;GO SEE IF MORE MEMORY TO TEST
5771  021430  004737  032010      50$:    JSR     PC,GETXDP       ;RESTORE XXDP LOADER
5772  021434  105737  003143              TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
5773  021440  001402                      BEQ     54$             ;BR IF NOT
5774  021442  005037  172516              CLR     @#SR3           ;DISABLE UNIBUS MAP
5775  021446                      54$:
5776
5777
5778
5779                            ;;*****************************************************************
5780                            ;*TEST 3         NPR/MEMORY BLOCK ADDRESSING TEST
5781                            ;*IN THIS TEST, MEMORY BLOCK ADDRESSING CAPABILITY DURING RK06
```

```
                                        ;*NPR'S IS TESTED.
                                        ;*BEFORE TESTING BEGINS, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS RWBUF.
                                        ;*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.
                                        ;;*****************************************************************
021446  000004                  TST3:   SCOPE
021450  012737  000003  001324          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
021456  004737  033640                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021462  004737  033706                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
021466  000137  022006                  JMP     TST4            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
021472  004737  032250                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
021476  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;SAVE XXDP LOADER, IF PRESENT
021502  004737  031730                  JSR     PC,FNDXDP       ;COMPUTE STARTING ADRS OF XXDP
021506  012737  065636  005542          MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADDRESS
021514  005037  005544                  CLR     XDPSAV+2
021520  004737  032002                  JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
021524  112737  000001  003133          MOVB    #1,XOVLAD       ;SET INDICATORS
021532  112737  000001  003134          MOVB    #1,XDPSVD
021540  012737  177777  005536          MOV     #-1,PATRN       ;SET FLAG FOR SVPRMS
                                        ;INIT PARAMETERS, WRITE AND READ BACK ALL MEMORY BLOCKS
021546  005037  005524          2$:     CLR     SELECT          ;INIT COMPARE FLAG
021552  004737  040612                  JSR     PC,SETUPM       ;INIT PARAMS
021556  012704  000001                  MOV     #1,R4           ;INIT DATA WORD
021562  004737  05236           4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WRD CNT FOR THIS MA
021566  005701                          TST     R1              ;SEE IF ALL MEMORY TESTED YET
021570  100473                          BMI     30$             ;BR IF DONE WRITING AND READING
021572  005065  000012                  CLR     P.WC(R5)        ;INIT WRD CNT TO 65,536(DEC)
021576  005701                          TST     R1              ;SEE IF WRD CNT SHOULD BE 65536
021600  001003                          BNE     6$              ;BR IF YES
021602  005400                          NEG     R0
021604  010065  000012                  MOV     R0,P.WC(R5)     ;SET WORD COUNT
021610  013765  005600  000010  6$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
021616  013701  005602                  MOV     MA+2,R1         ;GET MA BITS 16-21
021622  042701  177774                  BIC     #177774,R1      ;MASK FOR LO 2 BITS
021626  150165  000007                  BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
021632  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
021636  100010                          BPL     14$             ;BR IF NO
                                        ;SET UP MEMORY MANAGEMENT
021640  013737  005600  005604          MOV     MA,PMA          ;SET BUFFER ADDRESS
021646  013737  005602  005606          MOV     MA+2,PMA+2
021654  004737  031606                  JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBUS MAP
021660  005737  005524          14$:    TST     SELECT          ;SEE IF COMPARES SHOULD BE DONE YET
021664  001026                          BNE     20$             ;BR IF YES
                                        ;WRITE AND READ THIS MEM BLK ON RK06
021666  004737  042020          16$:    JSR     PC,SVPRMS       ;SAVE PARAMS FOR THIS XFER
```

```
5838  021672  010403                          MOV     R4,R3           ;GET DATA WORD
5839  021674  004737  037670                  JSR     PC,LDMEM2       ;LOAD MEMORY BLK WITH THIS WORD
5840  021700  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
5841  021706  004737  042132                  JSR     PC,TRNSFR       ;WRITE THE DATA
5842  021712  005003                          CLR     R3
5843  021714  004737  037670                  JSR     PC,LDMEM2       ;ZERO THE BLK IN MEMORY
5844  02172C  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
5845  021726  004737  042132                  JSR     PC,TRNSFR       ;READ THIS DATA BACK
5846  021732  005204                          INC     R4              ;INCREMENT THE DATA WORD
5847  021734  004737  041020                  JSR     PC,INCRMA       ;COMPUTE NEW MA TO TRY
5848  021740  000710                          BR      4$              ;GO WRITE AND READ NEXT BLK
5849                          ;PERFORM SOFTWARE COMPARES OF ALL MEMORY BLOCKS
5850  021742  010403          20$:    MOV     R4,R3           ;GET DATA WORD
5851  021744  004737  040034                  JSR     PC,CKMEM2       ;COMPARE THIS BLK TO GOOD DATA WORD
5852  021750  005204                          INC     R4              ;INCREMENT THE DATA WORD
5853  021752  004737  041020                  JSR     PC,INCRMA       ;GET NEW MA TO TRY
5854  021756  000701                          BR      4$              ;GO COMPARE NEXT MEM BLK
5855
5856  021760  005137  005524          30$:    COM     SELECT          ;COMPLEMENT THE FLAG
5857  021764  001401                          BEQ     50$             ;BR IF ALL DONE NOW
5858  021766  000671                          BR      2$              ;BR IF COMPARES SHOULD BE DONE NOW
5859  021770  004737  032010          50$:    JSR     PC,GETXDP       ;RESTORE SAVED XXDP
5860  021774  105737  003143                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
5861  022000  001402                          BEQ     54$             ;BR IF NOT
5862  022002  005037  172516                  CLR     @#SR3           ;DISABLE UNIBUS MAP
5863  022006                          54$:
5864
5865
5866
5867                          ;;*****************************************************************
5868                          ;*TEST 4        NPR/MEMORY DATA PATTERN TEST
5869                          ;*BEFORE TESTING, THE XXDP LOADER IS SAVED, IF PRESENT, AT ADRS
5870                          ;*RWBUF. THEN, ALL PHYSICAL MEMORY LOCATIONS , STARTING AT THE
5871                          ;*FIRST ADRS OF THE NEXT 1K MEMORY BLOCK BEYOND RWBUF+6000(OCT),
5872                          ;* ARE EXERCISED WITH UP TO 16 DATA PATTERNS ,
5873                          ;* CHOSEN IN PARAMETER PT. (THESE ARE THE SAME PATTERNS
5874                          ;*PROVIDED FOR USE IN THE READ/WRITE DATA TEST). IF PT = 0, THE
5875                          ;*DATA PATTERNS WILL DEFAULT TO PATS 08,09,10,AND 11.
5876                          ;*MEMORY IS EXERCISED IN BLOCKS OF UP TO 64K WORDS, STARTING AT
5877                          ;*THE FIRST ADDRESS OF THE NEXT 1K MEMORY BLOCK BEYOND THE BUFFER.
5878                          ;*EACH BLOCK IS LOADED WITH DATA, AND WRITTEN ONTO DISK AT ADDRESS
5879                          ;*FC,FT,FS (SCALED TO AVOID PACK OVERFLOW OR DESTRUCTION OF THE BAD
5880                          ;*SECTOR FILE). THEN THE MEMORY BLOCK IS ZEROED AND READ BACK AND
5881                          ;*COMPARED. THIS IS DONE FOR EACH OF THE DATA PATTERNS CHOSEN,
5882                          ;*POSSIBLY INCLUDING THE USER-DEFINED PATTERN 15, IF SPECIFIED.
5883                          ;;*****************************************************************
5884  022006  000004          TST4:   SCOPE
5885  022010  012737  000004  001324          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5886  022016  004737  033640                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5887  022022  004737  033706                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5888  022026  000137  022356                  JMP     TST5            ;JUMP TO NEXT TEST
5889                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5890  022032  004737  032250                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5891  022036  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5892                          ;SAVE XXDP LOADER, IF PRESENT
5893  022042  004737  031730                  JSR     PC,FNDXDP       ;COMPUTE STARTING ADR OF XXDP
```

```
5894  022046  012737  065636  005542          MOV     #RWBUF,XDPSAV   ;SET XXDP SAVE AREA ADR
5895  022054  005037  005544                  CLR     XDPSAV+2
5896  022060  004737  032002                  JSR     PC,SAVXDP       ;GO SAVE XXDP LOADER
5897  022064  112737  000001  003133          MOVB    #1,XOVLAD       ;SET INDICATORS
5898  022072  112737  000001  003134          MOVB    #1,XDPSVD
5899  022100  004737  036002                  JSR     PC,LODP14       ;GENERATE PSEUDO-RAND PAT 14
5900  022104  013737  005700  005536          MOV     PT,PATRN        ;GET A COPY OF  PT
5901  022112  001003                          BNE     2$              ;BR IF PT NON-ZERO
5902  022114  012737  007400  005536          MOV     #7400,PATRN     ;SET DEFLT PATRNS = PAT 8,9,10,11
5903                        ;INIT PARAMETERS, BEGIN TESTING
5904  022122  004737  040612          2$:     JSR     PC,SETUPM       ;INIT PARAMS
5905  022126  004737  035236          4$:     JSR     PC,MXWRDC       ;COMPUTE MAX WORD COUNT FOR THIS MA
5906  022132  005701                          TST     R1              ;SEE IF ALL MEMORY TESTED YET
5907  022134  100501                          BMI     50$             ;BR IF ALL DONE
5908  022136  005065  000012                  CLR     P.WC(R5)        ;INIT WORD COUNT TO 65,536(DEC)
5909  022142  005701                          TST     R1              ;SEE IF WRD CNT SHOULD BE 65,536
5910  022144  001003                          BNE     6$              ;BR IF YES
5911  022146  005400                          NEG     R0
5912  022150  010065  000012                  MOV     R0,P.WC(R5)     ;SET WORD COUNT
5913  022154  013765  005600  000010  6$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
5914  022162  013701  005602                  MOV     MA+2,R1         ;GET MA BITS 16-21
5915  022166  042701  177774                  BIC     #177774,R1      ;MASK FOR LO 2 BITS
5916  022172  150165  000007                  BISB    R1,P.BAHI(R5)   ;SET BA BITS 16,17
5917  022176  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
5918  022202  100010                          BPL     14$             ;BR IF NOT
5919                        ;SET UP MEM MGT
5920  022204  013737  005600  005604          MOV     MA,PMA          ;SET BUFFER ADDRESS
5921  022212  013737  005602  005606          MOV     MA+2,PMA+2
5922  022220  004737  031606                  JSR     PC,PREPAR       ;PREPARE MEM MGT REG'S AND UNIBUS MAP
5923  022224  012704  000001          14$:    MOV     #1,R4           ;SET PATTERN BIT POINTER
5924  022230  030437  005536          16$:    BIT     R4,PATRN        ;SEE IF THIS PATTERN IS CHOSEN
5925  022234  001003                          BNE     20$             ;BR IF YES
5926  022236  006304                  18$:    ASL     R4              ;SHIFT TO  POINT TO NEXT PATTERN
5927  022240  001434                          BEQ     24$             ;BR IF NO MORE LEFT TO CHECK
5928  022242  000772                          BR      16$             ;GO CHECK FOR ANOTHER PATTERN
5929  022244  010401                  20$:    MOV     R4,R1           ;GET A COPY OF BIT POINTER
5930                        ;PERFORM THE TESTS
5931  022246  012737  022246  001110  22$:    MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
5932  022254  004737  033640                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5933  022260  004737  042020                  JSR     PC,SVPRMS       ;SAVE THE PARAMS FOR THIS XFER
5934  022264  004737  036216                  JSR     PC,LODBUF       ;LOAD THIS MEM BLK WITH SELECTED DATA
5935  022270  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
5936  022276  004737  042132                  JSR     PC,TRNSFR       ;WRITE THE DATA
5937  022302  005003                          CLR     R3              ;SET DATA WORD = 0
5938  022304  004737  037670                  JSR     PC,LDMEM2       ;LOAD MEM BLK WITH ALL ZEROS
5939  022310  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
5940  022316  004737  042132                  JSR     PC,TRNSFR       ;READ THE DATA FROM DISK
5941  022322  004737  036444                  JSR     PC,CMPBUF       ;PERFORM SOFTWARE COMPARE OF DATA
5942  022326  104411                          SCOPER                  ;CHECK FOR INTERNAL LOOP ON ERROR
5943  022330  000742                          BR      18$             ;CHECK FOR  NEXT PATTERN
5944                        ;GET NEW MA FOR NEXT TRANSFER
5945  022332  004737  041020          24$:    JSR     PC,INCRMA       ;COMPUTE NEXT MA
5946  022336  000673                          BR      4$              ;GO SEE IF MORE MEMORY TO TEST
5947  022340  004737  032010          50$:    JSR     PC,GETXDP       ;RESTORE XXDP LOADER
5948  022344  105737  003143                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
5949  022350  001402                          BEQ     54$             ;BR IF NOT PRESENT
```

```
                                                    CLR     @#SR3           ;DISABLE UNIBUS MAP
                                            54$:
                                    ;;**************************************************************
                                    ;*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.

                                    ;;**************************************************************
022356  000004                      TST5:   SCOPE
022360  012737  000005  001324              MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
022366  004737  033640              JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
022372  004737  033706              JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022376  000137  022662              JMP     TST6            ;JUMP TO NEXT TEST
                                    ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022402  004737  032250              JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022406  004737  030536              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                    ;INITIALIZE PARAMETERS
022412  013765  005660  000002      MOV     FC,P.CYLN(R5)   ;SET CYL = FC
022420  113765  005664  000005      MOVB    FT,P.TRCK(R5)   ;SET TRACK = FT
022426  105065  000004              CLRB    P.SECT(R5)      ;SET SECTOR = 0
022432  012765  165000  000012      MOV     #-13000,P.WC(R5) ;INIT WORD COUNT FOR FULL TRACK
022440  105737  003125              TSTB    FORMAT          ;DETERMINE THE FORMAT
022444  001403                      BEQ     4$              ;BR IF 22 SECTORS
022446  012765  166000  000012      MOV     #-12000,P.WC(R5) ;SET WORD COUNT FOR FULL TRACK
022454  052765  100000  000014  4$: BIS     #DTBAII,P.PRST(R5) ;SET BUS ADDRESS INCREMENT INHIBIT
                                    ;WRITE DATA WITH ALLOWABLE UNIBUS CONTENTION, AND WRITE CHECK IT
022462  012700  006776              MOV     #PAT15,R0       ;SET DATA PATTERN ADDRESS
022466  005737  005700              TST     PT              ;SEE IF USER-DEFINED PATTERN 15 SELECTED
022472  100402                      BMI     14$             ;BR IF YES
```

```
 6006  022474  012700  006676                 MOV     #PAT13,R0          ;SET DATA PATTERN 13 ADDRESS
 6007  022500  012701  000020          14$:    MOV     #16.,R1            ;SET COUNTER = 16
 6008  022504  012703  000070                  MOV     #70,R3             ;SET NON-EXISTENT MEMORY TIMER
 6009  022510  010065  000010          16$:    MOV     R0,P.BALO(R5)      ;SET BA BITS
 6010  022514  105037  003141                  CLRB    DLTFLG             ;CLEAR DATA LATE FLAG
 6011  022520  112737  000001  003142          MOVB    #1,NORTRY          ;SET "NO-RETRY" FLAG
 6012  022526  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
 6013  022534  004737  040716                  JSR     PC,REFNEM          ;WRITE DATA DURING NEM REF.
 6014  022540  105737  003141                  TSTB    DLTFLG             ;SEE IF DATA LATE ERROR OCCURRED
 6015  022544  001011                          BNE     18$                ;BR IF NOT
 6016  022546  032737  000002  005474          BIT     #BSERR,RECODE      ;SEE IF BAD SECTOR ERROR
 6017  022554  001042                          BNE     24$                ;BR IF YES
 6018  022556  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
 6019  022564  004737  042510                  JSR     PC,DRVCAL          ;DO WRITE CHECK
 6020                          ;READ DATA WITH ALLOWABLE UNIBUS CONTENTION, AND COMPARE LAST WORD
 6021  022570  012765  065636  000010  18$:    MOV     #RWBUF,P.BALO(R5) ;SET BA = RWBUF ADDRESS
 6022  022576  005037  065636                  CLR     RWBUF              ;CLEAR THE BUFFER WORD
 6023  022602  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 6024  022610  004737  040716                  JSR     PC,REFNEM          ;READ DATA DURING NEM REF.
 6025  022614  105737  003141                  TSTB    DLTFLG             ;SEE IF DATA LATE ERROR OCCURRED
 6026  022620  001014                          BNE     20$                ;BR IF YES
 6027  022622  022037  065636                  CMP     (R0)+,RWBUF        ;COMPARE LAST DATA WORD
 6028  022626  001411                          BEQ     20$                ;BR IF EQUAL
 6029  022630  004737  043676                  JSR     PC,REPSUP          ;GATHER STATUS FOR PRINTOUT
 6030  022634  016037  177776  001174          MOV     -2(R0),$REG5       ;GOOD DATA WORD
 6031  022642  013737  065636  001176          MOV     RWBUF,$REG6        ;BAD DATA WORD
 6032  022650  104110                          ERROR   110                ;"READ ERROR WHILE BAI SET"
 6033                          ;SET UP FOR NEXT PASS THROUGH LOOP
 6034  022652  005301                  20$:    DEC     R1                 ;SEE IF ALL DONE YET
 6035  022654  001402                          BEQ     24$                ;BR IF YES
 6036  022656  000137  022510                  JMP     16$                ;JUMP TO CONTINUE TESTING
 6037  022662                          24$:
 6038
 6039
 6040
 6041                          ;;*****************************************************************
 6042                          ;*TEST 6        MULTI-DRIVE INTERFERENCE TEST
 6043                          ;*THIS TEST PERFORMS MULTI-DRIVE POSITIONING OPERATIONS, WHILE A
 6044                          ;*LARGE DATA TRANSFER IS IN PROGRESS ON THE DRIVE UNDER TEST, FOR
 6045                          ;*THE PURPOSE OF DETECTING PROBLEMS OF CONCURRENT DRIVE OPERATION.
 6046                          ;*THE TEST IS RUN ONLY IF THERE IS MORE THAN 1 DRIVE ON THE SUBSYSTEM.
 6047                          ;*
 6048                          ;*THE TEST PROCEEDS AS FOLLOWS :  IT IS FIRST DETERMINED WHICH DRIVE(S)
 6049                          ;*(BESIDE THE ONE UNDER TEST) EXIST AND ARE OPERATIONAL. THESE DRIVES
 6050                          ;*ARE THEN CLEARED AND A SEEK TO CYL 0 IS DONE ON EACH.NEXT, A SEEK IS
 6051                          ;*DONE TO CYLINDER FC (SCALED, IF NECESSARY,) ON THE DRIVE UNDER
 6052                          ;*TEST. A SEEK IS NOW BEGUN ON EACH OF THE OTHER DRIVES, EACH
 6053                          ;*TO A DIFFERENT PSEUDO-RANDOM CYLINDER ADDRESS. THIS SHOULD REQUIRE
 6054                          ;*FROM 10-67 MILLI-SEC TO COMPLETE. AS SOON AS THESE SEEKS ARE BEGUN,
 6055                          ;*A WRITE DATA COMMAND IS BEGUN ON THE DRIVE UNDER TEST AT THE CURRENT
 6056                          ;*CYLINDER (FC), SECTOR 0, TRACK 0, WITH THE BUS ADDRESS INCREMENT
 6057                          ;*INHIBIT BIT SET IN THE CONTROLLER, AND WITH A WORD COUNT OF 15,616
 6058                          ;*(DEC) WORDS IF 20 SECTOR FORMAT OR 17,152(DEC) WORDS IF 22 SECTOR
 6059                          ;*FORMAT. THIS WORD COUNT AMOUNTS TO A FULL CYLINDER + 1 SECTOR OF DATA.
 6060                          ;*THIS TRANSFER REQUIRES FROM 101-126 MILLI-SEC TO COMPLETE, AND IN
 6061                          ;*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.
                                        ;;*****************************************************************
022662  000004                  TST6:   SCOPE
022664  012737  000006  001324          MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
022672  004737  033640                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
022676  004737  033706                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022702  000137  023702                  JMP     $EOP            ;JUMP TO END-OF-PASS ROUTINE
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022706  004737  032250                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022712  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;SEARCH BAD SECTOR FILES FOR 2 ADJACENT ERROR-FREE CYLS TO USE
022716  013703  005660                  MOV     FC,R3           ;GET FC
022722  005004                  40$:    CLR     R4
022724  012700  003234          42$:    MOV     #BSSOFT+10,R0   ;POINTER TO SOFTWARE BSF
022730  020310                  52$:    CMP     R3,(R0)         ;SEE IF THIS CYL LISTED
022732  001006                          BNE     44$             ;BR IF NOT
022734  005203                          INC     R3              ;INCR CYL
022736  020337  020000                  CMP     R3,LSTCYL       ;SEE IF CYL < 632/1456
022742  103767                          BLO     40$             ;BR IF YES
022744  000137  023702                  JMP     MULTI1          ;GO SKIP THIS TEST
022750  062700  000004          44$:    ADD     #4,R0           ;INCR BSF POINTER
022754  020027  004224                  CMP     R0,#BSFACT      ;SEE IF STILL IN SOFTWARE BSF
022760  001002                          BNE     46$             ;BR IF YES
022762  062700  000010                  ADD     #10,R0          ;INCR POINTER FOR FACTORY BSF
022766  020027  005224          46$:    CMP     R0,#BSFACT+512. ;SEE IF END OF FACTORY BSF REACHED
022772  001356                          BNE     52$             ;BR IF NOT YET
022774  005704                          TST     R4              ;SEE IF CAME HERE BEFORE
022776  001003                          BNE     50$             ;BR IF YES
023000  005204                          INC     R4
023002  005203                          INC     R3              ;INC CYL
023004  000747                          BR      42$             ;CONTINUE LOOKING
023006  005303                  50$:    DEC     R3              ;DECR CYL
023010  010337  003202                  MOV     R3,CYL          ;GET CYL TO USE
                                        ;CHECK DRVLST FOR OTHER AVAILABLE DRIVES, AND RECALIBRATE THEM
023014  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
023022  005001                          CLR     R1              ;INIT MULTIPLE-DRIVES FLAG
023024  005000                          CLR     R0              ;INIT DRIVE NO. TO 0
023026  005065  000002                  CLR     P.CYLN(R5)      ;SET CYL = 0
023032  022700  000010          4$:     CMP     #10,R0          ;SEE IF ALL DRIVES CHECKED YET
023036  001415                          BEQ     8$              ;BR IF ALL DONE
023040  105760  005610                  TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
023044  001410                          BEQ     6$              ;BR IF NOT MARKED
023046  120037  005500                  CMPB    R0,DRIVE        ;SEE IF MARKED DRIVE IS DRIVE UNDER TEST
023052  001405                          BEQ     6$              ;BR IF YES
023054  110065  000000                  MOVB    R0,P.DRVN(R5)   ;SET DRIVE NO. PARAMETER
023060  004737  023564                  JSR     PC,TSTTYP       ;SEEK TO CYL 0 ON CURRENT DRIVE
023064  005201                          INC     R1              ;SET FLAG TO INDICATE MORE THAN 1 DRIVE
023066  005200                  6$:     INC     R0              ;INCR DRIVE NO.
023070  000760                          BR      4$              ;KEEP CHECKING DRIVES
023072  005701                  8$:     TST     R1              ;SEE IF MORE THAN 1 DRIVE PRESENT
023074  001002                          BNE     9$              ;BR IF YES, TO RUN TEST
023076  000137  023702                  JMP     MULTI1          ;NO, SKIP TEST
```

```
6118                                    ;SEEK TO CYLINDER CYL ON DRIVE UNDER TEST. SET PARAMS FOR DATA XFER
6119  023102  004737  032250            9$:     JSR     PC,INITSS       ;INIT S.S.
6120  023106  013765  003202  000002            MOV     CYL,P.CYLN(R5)  ;SET CYLNDR = CYL
6121  023114  005065  000004                    CLR     P.SECT(R5)      ;SET TRACK AND SECTOR = 0
6122  023120  012765  136400  000012            MOV     #-17152.,P.WC(R5) ;SET WC FOR 67(DEC) SECTORS
6123  023126  105737  003125                    TSTB    FORMAT          ;CHECK THE FORMAT
6124  023132  001403                            BEQ     12$             ;BR IF 22-SECTOR FORMAT
6125  023134  012765  141400  000012            MOV     #-15616.,P.WC(R5) ;SET WC FOR 61(DEC) SECTORS
6126  023142  012765  006676  000010  12$:     MOV     #PAT13,P.BALO(R5) ;SET BA BITS
6127  023150  052765  100000  000014            BIS     #DTBAII,P.PRST(R5) ;SET BUS ADDRESS INCREMENT INHIBIT
6128  023156  112765  000117  000001            MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
6129  023164  004737  042510                    JSR     PC,DRVCAL       ;SEEK TO CYL ON DRIVE UNDER TEST
6130                                    ;DO PSEUDO-RANDOM SEEKS ON ALL OTHER DRIVES
6131  023170  105037  003120                    CLRB    TSTING          ;INHIBIT ^C, ^Z ESCAPE
6132  023174  012701  000007                    MOV     #7,R1           ;SET LOOP COUNTER
6133  023200  012700  003204                    MOV     #CYLLST,R0      ;ADRS OF RAND CYL LIST
6134  023204  004737  035634            15$:    JSR     PC,RNDADR       ;GENERATE A PSEUDO-RANDOM CYL NO.
6135  023210  042737  177145  005504            BIC     #177145,CYLNDR  ;ALWAYS KEEP UNDER 632
6136  023216  013720  005504                    MOV     CYLNDR,(R0)+    ;SET CYL NO. IN TABLE
6137  023222  001002                            BNE     16$             ;BR IF NOT 0
6138  023224  005260  177776                    INC     -2(R0)          ;MAKE IT NON-ZERO
6139  023230  005301                    16$:    DEC     R1              ;DECREMENT LOOP COUNTER
6140  023232  001364                            BNE     15$             ;BR IF NOT DONE LOADING LIST YET
6141  023234  012701  003204                    MOV     #CYLLST,R1      ;GET ADRS OF RAND CYL TABLE
6142  023240  010537  023332                    MOV     R5,19$          ;SET PARAM BLK ADRS FOR DRIVER
6143  023244  005000                            CLR     R0              ;INIT DRIVE NO. TO 0
6144  023246  022700  000010            18$:    CMP     #10,R0          ;SEE IF ALL DRIVES CHECKED YET
6145  023252  001432                            BEQ     22$             ;BR IF YES
6146  023254  105760  005610                    TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE PRESENT
6147  023260  001425                            BEQ     20$             ;BR IF NO
6148  023262  120037  005500                    CMPB    R0,DRIVE        ;SEE IF THIS IS DRIVE UNDER TEST
6149  023266  001422                            BEQ     20$             ;BR IF YES
6150  023270  110065  000000                    MOVB    R0,P.DRVN(R5)   ;SET DRIVE NO. PARAMETER
6151  023274  012165  000002                    MOV     (R1)+,P.CYLN(R5) ;SET CYLINDER NO.
6152  023300  004737  042570                    JSR     PC,STRCMD       ;STORE PREV AND CURRENT CMNDS
6153  023304  105760  005620                    TSTB    DTYLST(R0)      ;SEE IF RK06
6154  023310  001404                            BEQ     1$              ;BR IF YES
6155  023312  152765  000004  000007            BISB    #B.CDT,P.CS1H(R5)       ;ELSE SET FOR RK07
6156  023320  000402                            BR      2$
6157  023322  105065  000007            1$:     CLRB    P.CS1H(R5)
6158  023326  004737  052562            2$:     JSR     PC,C.INIT       ;GO START SEEK ON THIS DRIVE
6159  023332  000000                    19$:    .WORD                   ;P.B. ADRS GOES HERE
6160  023334  005200                    20$:    INC     R0              ;INCR DRIVE NO.
6161  023336  000743                            BR      18$             ;BR TO SEEK ON NEXT DRIVE
6162                                    ;WRITE DATA ON DRIVE UNDER TEST, WITH BAI SET
6163  023340  112765  000123  000001    22$:    MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
6164  023346  113765  005500  000000            MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6165  023354  013765  003202  000002            MOV     CYL,P.CYLN(R5)  ;SET CYL NO.
6166  023362  012737  023614  003044            MOV     #MULHDL,A.NORM  ;SET SPECIAL DRIVER RETURN ADDRESS
6167  023370  113700  005500                    MOVB    DRIVE,R0
6168  023374  004737  023564                    JSR     PC,TSTTYP       ;WRITE 1 CYL + 1 SECTOR AT CYL FC
6169  023400  032737  000002  005474            BIT     #BSERR,RECODE   ;SEE IF BSE ERROR
6170  023406  001135                            BNE     MULTI1          ;BR IF YES
6171  023410  112765  000131  000001            MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6172  023416  004737  042510                    JSR     PC,DRVCAL       ;WRITE CHECK THE DATA WRITTEN
6173                                    ;CHECK ATTENTIONS AND CHECK CYLINDER ADDRESSES FROM ALL OTHER DRIVES
```

```
6174  023422  112737  000001  003120          MOVB    #1,TSTING          ;ALLOW ^C, ^Z ESCAPE
6175  023430  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
6176  023436  012701  003204                  MOV     #CYLLST,R1         ;ADRS OF RAND CYL LIST
6177  023442  005000                          CLR     R0                 ;INIT DRIVE NO.
6178  023444  022700  000010          26$:    CMP     #10,R0             ;SEE IF ALL DRIVES CHECKED YET
6179  023450  001514                          BEQ     MULTI1             ;BR IF YES
6180  023452  105760  005610                  TSTB    DRVLST(R0)         ;SEE IF THIS DRIVE MARKED IN LIST
6181  023456  001440                          BEQ     32$                ;BR IF NOT
6182  023460  120037  005500                  CMPB    R0,DRIVE           ;SEE IF THIS IS DRIVE UNDER TEST
6183  023464  001435                          BEQ     32$                ;BR IF YES
6184  023466  136037  003102  003200          BITB    I.DRV(R0),SAVWRD   ;SEE IF GOT ATT'N FROM THIS DRIVE
6185  023474  001003                          BNE     28$                ;BR IF YES
6186  023476  004737  043676                  JSR     PC,REPSUP          ;PREPARE STATUS FOR PRINTOUT
6187  023502  104113                          ERROR   113                ;"NO ATTENTION ON SEEK"
6188  023504  110065  000000          28$:    MOVB    R0,P.DRVN(R5)      ;SET THIS DRIVE NO.
6189  023510  004737  023564                  JSR     PC,TSTTYP          ;READ STATUS OF THIS DRIVE
6190  023514  016503  000052                  MOV     P.B10(R5),R3       ;GET STATUS BYTE B10
6191  023520  006003                          ROR     R3                 ;GET CYL ADRS RIGHT-JUSTIFIED
6192  023522  006003                          ROR     R3
6193  023524  006003                          ROR     R3
6194  023526  006003                          ROR     R3
6195  023530  042703  177000                  BIC     #177000,R3         ;CLEAR OFF UNUSED BITS
6196  023534  022103                          CMP     (R1)+,R3           ;COMPARE TO EXPECTED CYLINDER
6197  023536  001410                          BEQ     32$                ;BR IF EQUAL
6198  023540  016137  177776  001212          MOV     -2(R1),$REG14      ;GOOD CYL NO.
6199  023546  010337  001214                  MOV     R3,$REG15          ;BAD CYL NO.
6200  023552  004737  043676                  JSR     PC,REPSUP          ;GATHER STATUS FOR PRINTOUT
6201  023556  104114                          ERROR   114                ;"DRIVE'S CYLINDER INCORRECT"
6202  023560  005200                  32$:    INC     R0                 ;INCREMENT DRIVE NO.
6203  023562  000730                          BR      26$                ;BR IF NOT DONE WITH ALL DRIVES YET
6204
6205                                  ;THIS ROUTINE TESTS FOR THE DRIVE TYPE AND SETS P.CS1H ACCORDINGLY
6206
6207  023564  105760  005620          TSTTYP: TSTB    DTYLST(R0)         ;SEE IF RK06
6208  023570  001404                          BEQ     1$                 ;BR IF YES
6209  023572  152765  000004  000007          BISB    #B.CDT,P.CS1H(R5)          ;ELSE SETUP FOR RK07
6210  023600  000402                          BR      2$
6211  023602  105065  000007          1$:     CLRB    P.CS1H(R5)         ;SETUP FOR RK06
6212  023606  004737  042510          2$:     JSR     PC,DRVCAL
6213  023612  000207                          RTS     PC
6214
6215
6216                                  ;THIS IS THE NORMAL RETURN FROM THE DRIVER WHICH IS USED TO CHECK
6217                                  ;THE RESULTS OF MULTI-DRIVE OPERATIONS.
6218  023614  032737  000200  003000  MULHDL: BIT     #RDY,T.CS1         ;SEE IF CNTRLR RDY IS SET
6219  023622  001010                          BNE     6$                 ;BR IF RDY SET
6220  023624  004737  052162                  JSR     PC,I.CSTS          ;READ CONTROLLER REGISTERS
6221  023630  013765  003000  000016          MOV     T.CS1,P.CS1(R5)    ;GET CS1 BITS
6222  023636  004737  043676                  JSR     PC,REPSUP          ;PREPARE STATUS FOR PRINTOUT
6223  023642  104112                          ERROR   112                ;"INTRPT WHEN CNTRLR NOT RDY"
6224  023644  013737  003014  003200  6$:     MOV     T.ASOF,SAVWRD      ;SAVE ATT'N SUMMARY
6225  023652  000337  003200                  SWAB    SAVWRD             ;GET ATT'N BITS IN BITS 0-7
6226  023656  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6227  023664  012737  042700  003044          MOV     #ERRFRE,A.NORM     ;RESTORE NORMAL DRIVER RETURN ADDRESS
6228  023672  004737  042510                  JSR     PC,DRVCAL          ;CLEAR SUB-SYS
6229  023676  000137  042700                  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 APTPAS

023702                          $EOP:
023702  000004                          SCOPE
023704  005237  001330                  INC     $DEVCT          ;INCREMENT DEVICE COUNT FOR APT
023710  000137  017272                  JMP     NEWDRV          ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
023714                          DUNPAS: ;THIS IS TRULY THE END OF A PASS
023714  042777  000100  155222          BIC     #BIT6,@$TKS     ;DISABLE TTY KBD INTERRUPT
023722  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
023726  005037  001304                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
023732  005237  001326                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
023736  042737  100000  001326          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
023744  005327                          DEC     (PC)+           ;;LOOP?
023746  000001                  $EOPCT: .WORD   1
023750  003022                          BGT     $DOAGN          ;;YES
023752  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
023754  000001                  $ENDCT: .WORD   1
023756  023746                          $EOPCT
023760  104401  024025                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
023764  013746  001326                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
023770  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
023772  104401  024022                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
023776  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
024002  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
024004  000005                          RESET                   ;;CLEAR THE WORLD
024006  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
024010  000240                          NOP                     ;;SAVE ROOM
024012  000240                          NOP                     ;;FOR
024014  000240                          NOP                     ;;ACT11
024016                          $DOAGN:
024016  000137                          JMP     @(PC)+          ;;RETURN
024020  024042                  $RTNAD: .WORD   APTPAS
024022     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
024025     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
024032  050040  051501  020123
024040  000043
024042  122737  000001  001340  APTPAS: CMPB    #APTENV,$ENV    ;RUN UNDER APT ?
024050  001007                          BNE     2$      ;BRANCH IF NOT
024052  023727  001326  000002          CMP     $PASS,#2        ;TWO PASSES ?
024060  103403                          BLO     2$      ;BRANCH IF NOT
024062  005237  001102          1$:     INC     $TSTNM  ;INCREMENT TEST NUMBER
024066  000775                          BR      1$      ;WAITING LOOP
024070  000137  017260          2$:     JMP     NEWPAS  ;JMP TO THE MAIN PROGRAM

                                .SBTTL  RK06 HEAD ALIGNMENT AID
```

```
 6286  024074  012737  000340  177776  ASTART: MOV     #PR7,@#PS
 6287                                  .SBTTL  INITIALIZE THE COMMON TAGS
 6288                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 6289  024102  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 6290  024106  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 6291  024110  022706  001140                  CMP     #SWR,R6 ;;DONE?
 6292  024114  001374                          BNE     .-6             ;;LOOP BACK IF NO
 6293  024116  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 6294                                  ;;INITIALIZE A FEW VECTORS
 6295  024122  012737  057146  000020          MOV     #$SCOPE,@#.OTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 6296  024130  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 6297  024136  012737  056442  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 6298  024144  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 6299  024152  012737  060254  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 6300  024160  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 6301  024166  012737  057012  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 6302  024174  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 6303  024202  013737  023754  023746          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 6304  024210  005037  001304                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 6305  024214  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 6306  024220  112737  000001  0C1115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 6307  024226  012737  024226  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 6308  024234  012737  024234  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 6309                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 6310                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 6311  024242  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 6312  024246  012737  024302  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 6313  024254  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 6314  024262  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 6315  024270  022777  177777  154642          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 6316  024276  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 6317                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
 6318  024300  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
 6319  024302  012716  024310          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 6320  024306  000002                          RTI
 6321  024310  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 6322  024316  012737  000174  001142          MOV     #DISPREG,DISPLAY
 6323  024324  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 6324
 6325  024330  005037  001326                  CLR     $PASS           ;;CLEAR PASS COUNT
 6326  024334  132737  000200  001341          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
 6327  024342  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
 6328  024344  012737  001342  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
 6329  024352                          67$:
 6330  024352  005037  003150                  CLR     AUTOFG          ;CLEAR THE SPECIAL FLAG
 6331                                          .SBTTL  RK06 HEAD ALIGNMENT AID
 6332  024356  000005                          RESET                   ;RESET UNIBUS
 6333  024360  104401  007036                  TYPE    ,DZR6N          ;TYPE PROGRAM I.D. FOR PART 2
 6334  024364  104401  007050                  TYPE    ,SUBVER
 6335  024370  104401  007075                  TYPE    ,PART2
 6336  024374  012737  000176  001140          MOV     #SWREG,SWR      ;ADRS OF SOFTWARE SWR
 6337  024402  005077  154532                  CLR     @SWR            ;MAKE SWR BITS ALL 0
 6338  024406  012737  030376  000060          MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
 6339  024414  012737  000200  000062          MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
 6340  024422  013701  003040                  MOV     RKVEC,R1        ;GET ADDRESS OF VECTOR STORAGE
 6341  024426  012721  047262                  MOV     #I.INTR,(R1)+   ;SET IT TO INTERRUPT HNDLR
```

```
 6342  024432  012711  000340                MOV     #PR7,(R1)       ;SET INTERRUPT HANDLER PR7
 6343  024436  012737  000000  177776        MOV     #PR0,@#PS       ;ALLOW ALL INTERRUPTS
 6344  024444  012737  000000  005660        MOV     #0,FC           ;SET FIRST CYL = 0
 6345
 6346  024452                        GIVEID:
 6347  024452  012737  176543  054710        MOV     #176543,$HINUM  ;INITIALIZE PSEUDO-RANDOM NUMBERS
 6348  024460  012737  123456  054712        MOV     #123456,$LONUM
 6349  024466  105037  003116                CLRB    MDFLAG          ;CLEAR PARAM INP FLAG
 6350  024472  105037  003120                CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
 6351  024476  005037  005502                CLR     STALLS          ;DON'T STALL ON OPERATIONS
 6352  024502  012701  005224                MOV     #PRVCMD,R1      ;ZERO OUT PREV COMMAND
 6353  024506  012700  000006                MOV     #6,R0
 6354  024512  005021                42$:    CLR     (R1)+
 6355  024514  005300                DEC     R0
 6356  024516  001375                BNE     42$
 6357  024520  105037  003126                CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
 6358  024524  104401  011324                TYPE    ,IDENT          ;TYPE PROGRAM IDENTIFICATION
 6359  024530  105737  003145                TSTB    HLPOVL          ;SEE IF HELP FILE OVERLAID
 6360                                ;       BNE     ASKMOD          ;BR IF YES
 6361  024534  001011                        BNE     ASKTYP          ;ASK FOR ALL DRIVE TYPES
 6362  024536  104401  011370                TYPE    ,FORHLP         ;ASK IF HELP DESIRED
 6363  024542  104406                        RDCHR                   ;READ RESPONSE
 6364  024544  112601                        MOVB    (SP)+,R1        ;GET CHARACTER
 6365  024546  122701  000015                CMPB    #015,R1         ;SEE IF <CR> TYPED
 6366                                ;       BEQ     ASKMOD          ;BR IF NO HELP NEEDED
 6367  024552  001402                        BEQ     ASKTYP          ;ASK FOR ALL DRIVE TYPES
 6368  024554  104401  066255                TYPE    ,HLPFIL         ;HELP REQUESTED- TYPE INFO.
 6369
 6370  024560  104401  001315        ASKTYP: TYPE    ,$CRLF          ;TO ENTER ALL DRIVE TYPES
 6371  024564  104401  012513                TYPE    ,DTYMSG         ;MESSAGE TO ENTER DRIVE TYPE
 6372  024570  005000                        CLR     R0              ;START FROM DRIVE 0
 6373  024572  104401  001315        1$:     TYPE    ,$CRLF          ;CR-LF
 6374  024576  105037  007467                CLRB    BADDRV+6        ;CLEAR THE LOC 6 FOR TYPE
 6375  024602  104401  007461                TYPE    ,BADDRV         ;DRIVE ...
 6376  024606  010046                        MOV     R0,-(SP)        ;DRIVE NUMBER ON STACK
 6377  024610  104402                        TYPOC                   ;TYPE THE DRIVE NUMBER
 6378  024612  104401  012535                TYPE    ,SOR7           ;ENTER 6 OR 7
 6379  024616  004737  030536                JSR     PC,PREPKB       ;READ THE NUMBER
 6380  024622  005737  005522        2$:     TST     INTCHR          ;ANY THING READ IN YET ?
 6381  024626  001775                        BEQ     2$              ;BRANCH IF NOT
 6382  024630  105060  005620                CLRB    DTYLST(R0)      ;ASSUME IT'S A RK06
 6383  024634  013701  005522                MOV     INTCHR,R1       ;READ THE CH
 6384  024640  022701  000067                CMP     #'7,R1          ;IT'S A RK07 ?
 6385  024644  001003                        BNE     3$              ;BRANCH IF NOT
 6386  024646  112760  000004  005620        MOVB    #B.CDT,DTYLST(R0)       ;SET RK07 FLAG
 6387  024654  005200                3$:     INC     R0              ;TO NEXT DRIVE
 6388  024656  022700  000007                CMP     #7,R0           ;ALL 8 DRIVES ARE SET UP YET ?
 6389  024662  103343                        BHIS    1$              ;BRANCH IF NOT
 6390
 6391                                ;DETERMINE DESIRED OPERATIONAL MODE
 6392  024664  104401  011425        ASKMOD: TYPE    ,TYPMOD         ;ASK FOR DESIRED OPERATIONAL MODE
 6393  024670  004737  030536                JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 6394  024674  005737  005522        1$:     TST     INTCHR          ;CHECK FOR TTY INPUT
 6395  024700  001775                        BEQ     1$              ;BR IF NO INPUT YET
 6396  024702  013701  005522                MOV     INTCHR,R1       ;GET INPUT CHAR INTO R1
 6397  024706  022701  000022                CMP     #022,R1         ;SEE IF (^R) TYPED
```

```
6398  024712  001657                          BEQ     GIVEID          ;BR IF (^R), TO RESTART
6399  024714  022701  000115                  CMP     #'M,R1          ;IS IT MANUAL MODE ?
6400  024720  001002                          BNE     2$              ;BR IF NOT MANUAL
6401  024722  000137  024746                  JMP     MANUAL          ;JUMP TO MANUAL MODE ROUTINE
6402  024726  022701  000101          2$:     CMP     #'A,R1          ;IS IT AUTO MODE ?
6403  024732  001002                          BNE     3$              ;BR IF  NOT AUTO
6404  024734  000137  026104                  JMP     AUTO            ;JUMP TO AUTO MODE ROUTINE
6405  024740  004737  030556          3$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6406  024744  000747                          BR      ASKMOD          ;BR TO  ASK  FOR MODE AGAIN
6407
6408
6409
6410                                  ;MANUAL SELECT MODE ROUTINE
6411  024746                          MANUAL:
6412  024746  005037  003150                  CLR     AUTOFG          ;CLEAR SPECIAL FLAG
6413  024752  104401  011466                  TYPE    ,TYPMAN         ;TYPE "MANUAL SELECT MODE"
6414  024756  104401  011521          ASKDRV: TYPE    ,ENTDRV         ;ASK FOR DRIVE  NUMBER
6415  024762  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR KBD  INPUT
6416  024766  005737  005522          1$:     TST     INTCHR          ;SEE IF ANY INPUT
6417  024772  001775                          BEQ     1$              ;BR IF NONE TO CHECK AGAIN
6418  024774  013701  005522                  MOV     INTCHR,R1       ;GET CHARACTER INTO R1
6419  025000  020127  000022                  CMP     R1,#022         ;TEST FOR  (^R) TYPED
6420  025004  001622                          BEQ     GIVEID          ;BR IF (^R) TO RESTART
6421  025006  020127  000060                  CMP     R1,#'0          ;COMPARE TO ASCII  0
6422  025012  002512                          BLT     BAIFLU          ;BR IF <0 (BAD INPUT)
6423  025014  020127  000067                  CMP     R1,#'7          ;COMPARE TO ASCII  7
6424  025020  003107                          BGT     BAIFLU          ;BR IF  >7 (BAD INPUT)
6425  025022  042701  000060                  BIC     #'0,R1          ;STRIP  ASCII BITS
6426  025026  010137  005500                  MOV     R1,DRIVE        ;STORE DRIVE NUMBER
6427  025032  013700  005500                  MOV     DRIVE,R0        ;AND SET UP TO SIZE DRIVE
6428  025036  116037  005620  003124          MOVB    DTYLST(R0),TYPFMT ;LOAD THE FLAG FOR 6 OR 7
6429  025044  012737  000624  017776          MOV     #624,LCM6
6430  025052  012737  000631  005736          MOV     #631,PRMLIM+2
6431  025060  012737  000632  005662          MOV     #632,LC
6432  025066  012737  000632  005710          MOV     #632,PRDFLT+2
6433  025074  012737  000632  005742          MOV     #632,PRMLIM+6
6434  025102  012737  000400  020002          MOV     #400,HOLD1
6435  025110  012737  000025  020004          MOV     #25,HOLD3
6436  025116  012737  010025  020006          MOV     #10025,HOLD4
6437  025124  012737  000632  020000          MOV     #632,LSTCYL     ;LOAD THE LAST CYLINDER NUMBER
6438  025132  105737  003124                  TSTB    TYPFMT          ;IS AN RK07 ?
6439  025136  001433                          BEQ     9$              ;BRANCH IF NOT
6440  025140  012737  001450  017776          MOV     #1450,LCM6
6441  025146  012737  001455  005736          MOV     #1455,PRMLIM+2
6442  025154  012737  001456  005662          MOV     #1456,LC
6443  025162  012737  001456  005710          MOV     #1456,PRDFLT+2
6444  025170  012737  001456  005742          MOV     #1456,PRMLIM+6
6445  025176  012737  002025  020004          MOV     #2025,HOLD3
6446  025204  012737  012025  020006          MOV     #12025,HOLD4
6447  025212  012737  001000  020002          MOV     #1000,HOLD1
6448  025220  012737  001456  020000          MOV     #1456,LSTCYL    ;OTHERWISE SET UP THE LAST CYLINDER
6449  025226  000240                  9$:     NOP
6450                                  ;       MOVB    #-1,HDALI       ;SET PROGRAM STATUS FLAG
6451                                  ;       JMP     @#TYPED         ;GET DRIVE STATISTICS
6452                                  ;INTGO: CLR     HDALI           ;HOUSEKEEP
6453  025230  013737  020000  005662          MOV     LSTCYL,LC       ;AND SETUP MAX CYLINDER CNT
```

```
 6454  025236  000403                          BR      BAIA            ;PROCEED WITH DESIRED DRIVE
 6455  025240  004737  030556          BAIFLU: JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6456  025244  000644                          BR      ASKDRV          ;GO BACK TO ASK AGAIN
 6457  025246  004737  032250          BAIA:   JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
 6458                                  ;CHECK DESIRED DRIVE FOR PROPER STATUS
 6459  025252  004737  033220                  JSR     PC,CHKDRV       ;CHECK DRIVE FOR ON-LINE,READY,WRITE LOCK
 6460  025256  024756                          ASKDRV                  ;ADDRESS OF ERROR RETURN FOR CHKDRV
 6461  025260  004737  035312                  JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
 6462                                  ;SET UP DESIRED SUB-MODE OF OPERATION
 6463  025264                          SUBMOD:
 6464  025264  105037  011323                  CLRB    VERIFY          ;CLEAR VERIFY MODE FLAG
 6465  025270  104401  012^ 2                  TYPE    ,ENTMOD         ;ASK FOR ALIGN OR EXERCISE SUB-MODE
 6466  025274  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 6467  025300  005737  005522          1$:     TST     INTCHR          ;SEE IF ANY INPUT
 6468  025304  001775                          BEQ     1$              ;BR IF NONE TO CHECK A  .
 6469  025306  013701  005522                  MOV     INTCHR,R1       ;GET CHAR INTO R1
 6470  025312  020127  000022                  CMP     R1,#022         ;SEE IF (^R) TYPED
 6471  025316  001002                          BNE     2$              ;BR IF NOT (^R)
 6472  025320  000137  024452                  JMP     GIVEID          ;JUMP TO RESTART
 6473  025324  020127  000003          2$:     CMP     R1,#003         ;SEE IF (^C) TYPED
 6474  025330  001002                          BNE     3$              ;BR IF NOT (^C)
 6475  025332  000137  024756                  JMP     ASKDRV          ;JUMP TO ASK FOR DRIVE AGAIN
 6476  025336  020127  000101          3$:     CMP     R1,#'A          ;SEE IF ALIGNMENT REQUESTED
 6477  025342  001415                          BEQ     MANAL           ;BR IF MANUAL ALIGNMENT REQUESTED
 6478  025344  020127  000105                  CMP     R1,#'E          ;SEE IF EXERCISES REQUESTED
 6479  025350  001002                          BNE     4$              ;BR IF NOT EXERCISES
 6480  025352  000137  025650                  JMP     MANEX           ;JUMP TO DO MANUAL EXERCISES
 6481  025356  020127  000126          4$:     CMP     R1,#'V          ;SEE IF VERIFY REQUESTED
 6482  025362  001002                          BNE     6$              ;BR IF NOT VERIFY
 6483  025364  000137  025632                  JMP     MANVR           ;JUMP TO VERIFY MODE
 6484  025370  004737  030556          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6485  025374  000733                          BR      SUBMOD          ;ASK FOR SUB-MODE AGAIN
 6486
 6487
 6488
 6489                                  ;MANUAL ALIGNMENT MODE
 6490  025376                          MANAL:
 6491  025376  104401  012352                  TYPE    ,MANALN         ;REPORT MANUAL ALIGNMENT MODE
 6492                                  ;REQUEST AND CHECK DESIRED HEAD
 6493  025402  104401  012557          ASKHED: TYPE    ,ENTHED         ;ASK FOR DESIRED HEAD
 6494  025406  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 6495  025412  005737  005522          1$:     TST     INTCHR          ;SEE IF ANY INPUT
 6496  025416  001775                          BEQ     1$              ;BR IF NONE TO CHECK AGAIN
 6497  025420  013701  005522                  MOV     INTCHR,R1       ;GET CHAR INTO R1
 6498  025424  020127  000022                  CMP     R1,#022         ;SEE IF (^R)
 6499  025430  001007                          BNE     2$              ;BR IF NOT (^R)
 6500  025432  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6501  025440  004737  042510                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6502  025444  000137  024452                  JMP     GIVEID          ;JUMP TO RESTART
 6503  025450  020127  000003          2$:     CMP     R1,#003         ;SEE IF (^C)
 6504  025454  001007                          BNE     3$              ;BR IF NOT (^C)
 6505  025456  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6506  025464  004737  042510                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6507  025470  000137  024756                  JMP     ASKDRV          ;JUMP TO ASK FOR DRIVE AGAIN
 6508  025474  020127  000032          3$:     CMP     R1,#032         ;SEE IF (^Z)
 6509  025500  001007                          BNE     4$              ;BR IF NOT (^Z)
```

```
 6510  025502  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6511  025510  004737  042510                  JSR     PC,DRVCAL         ;RECALIBRATE DRIVE
 6512  025514  000137  025264                  JMP     SUBMOD            ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
 6513  025520  020127  000060          4$:     CMP     R1,#'0            ;COMPARE TRACK TO ASCII 0
 6514  025524  002410                          BLT     5$                ;BR IF <0 (BAD INPUT)
 6515  025526  020127  000062                  CMP     R1,#'2            ;COMPARE TRACK TO ASCII 2
 6516  025532  003005                          BGT     5$                ;BR IF >2 (BAD INPUT)
 6517  025534  042701  000060                  BIC     #'0,R1            ;STRIP ASCII BITS
 6518  025540  010137  005520                  MOV     R1,TRACK          ;STORE TRACK (HEAD) NUMBER
 6519  025544  000404                          BR      6$                ;GO SELECT HEAD
 6520  025546  004737  030556          5$:     JSR     PC,ECOBAD         ;ECHO BAD INPUT
 6521  025552  000137  025402                  JMP     ASKHED            ;ASK AGAIN FOR HEAD NUMBER
 6522                                  ;UNLOAD HEADS,AND WHEN READY, START SPINDLE AND SEEK TO ALN CYL
 6523  025556  105737  011323          6$:     TSTB    VERIFY            ;SEE IF VERIFY MODE
 6524  025562  001002                          BNE     8$                ;BR IF VERIFY
 6525  025564  004737  033534                  JSR     PC,WAIT4R         ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
 6526  025570  113765  005520  000005  8$:     MOVB    TRACK,P.TRCK(R5) ;SET DESIRED HEAD NUMBER
 6527  025576  004737  033460                  JSR     PC,ALNSEK         ;SEEK TO ALIGNMENT CYLINDER
 6528  025602  104401  012447                  TYPE    ,HEDPOS           ;TYPE HEADS POSITIONED MSG
 6529  025606  013701  005520                  MOV     TRACK,R1          ;GET BINARY TRACK NO.
 6530  025612  152701  000060                  BISB    #'0,R1            ;CONVERT TO ASCII
 6531  025616  110137  012643                  MOVB    R1,HEADNO         ;GET HEAD NO. INTO MSG BUFF.
 6532  025622  104401  012636                  TYPE    ,HEDSEL           ;TYPE HEAD SELECTED MSG
 6533  025626  000137  025402                  JMP     ASKHED            ;GO BACK TO ASK FOR NEW HEAD SELECT
 6534
 6535
 6536
 6537                                  ;MANUAL SELECT VERIFY MODE
 6538  025632  112737  000001  011323  MANVR:  MOVB    #1,VERIFY         ;SET VERIFY MODE FLAG
 6539  025640  104401  012412                  TYPE    ,MANVRF           ;REPORT MANUAL VERIFY MODE
 6540  025644  000137  025402                  JMP     ASKHED            ;GO ASK FOR DESIRED HEAD
 6541
 6542
 6543
 6544                                  ;MANUAL RANDOM SEEK EXERCISE ROUTINE
 6545  025650                          MANEX:
 6546  025650  004737  033534                  JSR     PC,WAIT4R         ;UNLOAD HEADS, WAIT FOR <R> TYPED, LOAD
 6547  025654  013701  005500                  MOV     DRIVE,R1          ;GET DRIVE NUMBER
 6548  025660  052701  000060                  BIS     #'0,R1            ;CONVERT TO ASCII
 6549  025664  110137  012344                  MOVB    R1,DRIEXR         ;PUT DRIVE NUMBER INTO OUTPUT BUFFER
 6550  025670  104401  012265                  TYPE    ,MANEXR           ;TYPE RANDOM SEEKS MSG
 6551  025674  012700  016514                  MOV     #RNDLNG,R0        ;INITIALIZE RANDOM SEEK COUNTER
 6552  025700  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 6553  025706  004737  030536          1$:     JSR     PC,PREPKB         ;PREPARE FOR KBD INPUT
 6554  025712  004737  035634          2$:     JSR     PC,RNDADR         ;SELECT RANDOM CYLINDER ADDRESS
 6555  025716  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
 6556  025724  004737  042510                  JSR     PC,DRVCAL         ;DO A RANDOM SEEK
 6557                                  ;CHECK FOR TTY INPUT DURING EXERCISES
 6558  025730  005737  005522                  TST     INTCHR            ;SEE IF ANY INPUT
 6559  025734  001016                          BNE     3$                ;BR IF A CHAR WAS TYPED
 6560  025736  005300                          DEC     R0                ;DECREMENT SEEK COUNTER
 6561  025740  001364                          BNE     2$                ;BR IF NOT DONE SEEKING YET
 6562  025742  042777  000100  153174          BIC     #BIT6,@$TKS       ;DISABLE KBD INTERRUPT
 6563  025750  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6564  025756  004737  042510                  JSR     PC,DRVCAL         ;RECALIBRATE DRIVE
 6565  025762  104401  001310                  TYPE    ,$BELL            ;RING BELL TO SIGNIFY END OF EXERCISES
```

```
 6566  025766  000137  025264                 JMP     SUBMOD          ;GO ASK FOR NEW MANUAL SUB-MODE
 6567  025772  013701  005522          3$:    MOV     INTCHR,R1       ;GET CHARACTER INTO R1
 6568  025776  020127  000022                 CMP     R1,#022         ;SEE IF (^R)
 6569  026002  001007                         BNE     4$              ;BR IF NOT (^R)
 6570  026004  112765  000113  000001         MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6571  026012  004737  042510                 JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6572  026016  000137  024452                 JMP     GIVEID          ;JUMP TO RESTART
 6573  026022  020127  000003          4$:    CMP     R1,#003         ;SEE IF (^C)
 6574  026026  001007                         BNE     5$              ;BR IF NOT (^C)
 6575  026030  112765  000113  000001         MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6576  026036  004737  042510                 JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6577  026042  000137  024756                 JMP     ASKDRV          ;JUMP TO ASK FOR NEW DRIVE NUMBER
 6578  026046  020127  000032          5$:    CMP     R1,#032         ;SEE IF (^Z)
 6579  026052  001007                         BNE     6$              ;BR IF NOT (^Z)
 6580  026054  112765  000113  000001         MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6581  026062  004737  042510                 JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6582  026066  000137  025264                 JMP     SUBMOD          ;JUMP TO ASK FOR NEW MANUAL SUB-MODE
 6583  026072  004737  030556          6$:    JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6584  026076  104401  012265                 TYPE    ,MANEXR         ;VERIFY STILL IN MANUAL EXERCISES
 6585  026102  000701                         BR      1$              ;BR TO CONTINUE EXERCISES
 6586
 6587
 6588
 6589                                  ;AUTO SELECT MODE ROUTINE
 6590  026104                          AUTO:
 6591  026104  012737  177777  003150         MOV     #-1,AUTOFG      ;SET SPECIAL FLAG
 6592  026112  004737  032250                 JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
 6593  026116  104401  012660                 TYPE    ,TYPAUT         ;TYPE "AUTO SELECT MODE"
 6594  026122  105037  011323          1$:    CLRB    VERIFY          ;CLEAR VERIFY MODE FLAG
 6595  026126  104401  012212                 TYPE    ,ENTMOD         ;ASK FOR ALIGN,VERIFY, OR EXERCISE
 6596  026132  004737  030536                 JSR     PC,PREPKB       ;PREPARE FOR KEYBOARD INPUT
 6597  026136  005737  005522          2$:    TST     INTCHR          ;SEE IF ANY INPUT
 6598  026142  001775                         BEQ     2$              ;BR IF NONE, TO CHECK AGAIN
 6599  026144  013701  005522                 MOV     INTCHR,R1       ;GET CHAR INTO R1
 6600  026150  020127  000022                 CMP     R1,#022         ;SEE IF (^R) TYPED
 6601  026154  001002                         BNE     3$              ;BR IF NOT (^R)
 6602  026156  000137  024452                 JMP     GIVEID          ;JUMP TO RESTART
 6603  026162  020127  000101          3$:    CMP     R1,#'A          ;SEE IF AUTO ALIGNMENT REQUESTED
 6604  026166  001417                         BEQ     AUTAL           ;BR IF AUTO ALIGNMENT REQUESTED
 6605  026170  020127  000105                 CMP     R1,#'E          ;SEE IF AUTO EXERCISES REQUESTED
 6606  026174  001002                         BNE     4$              ;BR IF EXERCISES NOT REQUESTED
 6607  026176  000137  027460                 JMP     AUTEX           ;JUMP TO DO AUTO EXERCISES
 6608  026202  020127  000126          4$:    CMP     R1,#'V          ;SEE IF VERIFY REQUESTED
 6609  026206  001004                         BNE     6$              ;BR IF NOT VERIFY
 6610  026210  112737  000001  011323         MOVB    #1,VERIFY       ;SET VERIFY MODE FLAG
 6611  026216  000403                         BR      AUTAL           ;PROCEED IN VERIFY MODE
 6612  026220  004737  030556          6$.    JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6613  026224  000736                         BR      1$              ;BR TO ASK FOR A OR E AGAIN
 6614
 6615
 6616
 6617                                  ;AUTO ALIGNMENT MODE
 6618  026226                          AUTAL:
 6619  026226  105737  011323                 TSTB    VERIFY          ;SEE IF AUTO VERIFY MODE
 6620  026232  001403                         BEQ     3$              ;BR IF NOT
 6621  026234  104401  012747                 TYPE    ,AUTVRF         ;REPORT AUTO VERIFY MODE
```

```
6622  026240  000402                          BR      4$
6623  026242  104401  012711          3$:     TYPE    ,AUTALN         ;TYPE ALIGNMENT MESSAGE
6624  026246  004737  030536          4$:     JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
6625  026252  005037  003024                  CLR     T.MR2           ;CLEAR MR2 STORAGE WORD
6626  026256  005037  005524                  CLR     SELECT          ;INITIALIZE AUTO-SELECTED DRIVE INDICATOR
6627  026262  012737  000377  005526          MOV     #377,ONLINE     ;INIT. OLD ONLINE DRIVE INDICATOR
6628  026270  012737  000377  005530          MOV     #377,NEWON      ;INIT. NEW ONLINE DRIVE INDICATOR
6629  026276  012737  000377  005534          MOV     #377,SCRACH     ;INITIALIZE LAST DRIVE INDICATOR
6630  026304  005737  005522          DRVLUP: TST     INTCHR          ;SEE IF ANY KBD INPUT
6631  026310  001423                          BEQ     6$              ;BR IF NO INPUT
6632  026312  113765  005534  000000          MOVB    SCRACH,P.DRVN(R5) ;DRIVE NO. FOR POSSIBLE RECALIBRATE
6633  026320  123727  005522  000032          CMPB    INTCHR,#032     ;SEE IF (^Z) TYPED
6634  026326  001002                          BNE     2$              ;BR IF NOT (^Z)
6635                                  ;       CMP     #377,SCRACH     ;SEE IF ANY DRIVE SELECTED YET
6636                                  ;       BEQ     1$              ;BR IF NONE
6637                                  ;       MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6638                                  ;       JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6639  026330  000137  026104          1$:     JMP     AUTO            ;RESTART AUTO MODE
6640  026334  123727  005522  000022  2$:     CMPB    INTCHR,#022     ;SEE IF (^R) TYPED
6641  026342  001002                          BNE     4$              ;BR IF NOT (^R)
6642                                  ;       CMP     #377,SCRACH     ;SEE IF ANY DRIVE SELECTED YET
6643                                  ;       BEQ     3$              ;BR IF NONE
6644                                  ;       MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
6645                                  ;       JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
6646  026344  000137  024452          3$:     JMP     GIVEID          ;RESTART ALIGNMENT AID
6647  026350  004737  030556          4$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
6648  026354  004737  030536                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE  KBD INPUT
6649  026360  005037  005500          6$:     CLR     DRIVE           ;ZERO THE DRIVE NUMBER
6650  026364  012700  000001                  MOV     #1,R0           ;INITIALIZE DRIVE BIT MASK
6651  026370  012737  030320  003046  INTG:   MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
6652  026376  050037  005530                  BIS     R0,NEWON        ;SET ON-LINE BIT FOR THIS DRIVE
6653                                  ;SELECT CURRENT DRIVE, CHECK FOR NON-EXISTENT DRIVE (NED) INDICATION
6654  026402  013701  005500                  MOV     DRIVE,R1        ;LOAD DRIVE NUMBER
6655  026406  116137  005620  003124          MOVB    DTYLST(R1),TYPFMT ;LOAD THE RK06-RK07 FLAG
6656  026414  012737  000624  017776          MOV     #624,LCM6
6657  026422  012737  000631  005736          MOV     #631,PRMLIM+2
6658  026430  012737  000632  005662          MOV     #632,LC
6659  026436  012737  000632  020000          MOV     #632,LSTCYL
6660  026444  012737  000632  005710          MOV     #632,PRDFLT+2
6661  026452  012737  000632  005742          MOV     #632,PRMLIM+6
6662  026460  012737  000400  020002          MOV     #400,HOLD1
6663  026466  012737  000025  020004          MOV     #25,HOLD3
6664  026474  012737  010025  020006          MOV     #10025,HOLD4
6665  026502  105737  003124                  TSTB    TYPFMT  ;IS AN RK07 ?
6666  026506  001433                          BEQ     9$              ;BRANCH IF NOT
6667  026510  012737  001450  017776          MOV     #1450,LCM6
6668  026516  012737  001455  005736          MOV     #1455,PRMLIM+2
6669  026524  012737  001456  005662          MOV     #1456,LC
6670  026532  012737  001456  020000          MOV     #1456,LSTCYL
6671  026540  012737  001456  005710          MOV     #1456,PRDFLT+2
6672  026546  012737  001456  005742          MOV     #1456,PRMLIM+6
6673  026554  012737  001000  020002          MOV     #1000,HOLD1
6674  026562  012737  002025  020004          MOV     #2025,HOLD3
6675  026570  012737  012025  020006          MOV     #12025,HOLD4
6676  026576  000240                  9$:     NOP                     ;EXIT
6677  026600  004737  032404                  JSR     PC,SCNDRV       ;CHECK THE SELECTED DRIVE
```

```
6678  026604  026646                          99$                     ;ERROR EXIT
6679  026606  000240                          NOP
6680  026610  000240                          NOP
6681  026612  000240                          NOP
6682  026614  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
6683  026622  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
6684  026630  000137  027110                  JMP     23$             ;BRANCH TO EXECUTE THE DRIVE
6685                                  ;       MOV     DRIVE,R0        ;GET THE SPECIFIED DRIVE
6686                                  ;       MOV     #-1,HDALI       ;SET UP FOR PROGRAM MARKER
6687                                  ;       JMP     @#TYPED         ;LOAD THE DEVICE TABLE
6688                                  ;INTGON:        CLR     HDALI           ;CLEAR THIS STATUS
6689  026634  013737  020000  005662          MOV     LSTCYL,LC       ;GET THE MAX CYLINDER SIZE FOR THIS DEVICE
6690  026642  004737  042510                  JSR     PC,DRVCAL       ;SELECT THIS DRIVE
6691  026646  022737  000007  005500  99$:    CMP     #7,DRIVE        ;SEE IF WE JUST CHECKED DRIVE 7
6692  026654  001405                          BEQ     10$             ;BR IF IT WAS DRIVE 7
6693  026656  005237  005500                  INC     DRIVE           ;ADD 1 TO  DRIVE NUMBER
6694  026662  000241                          CLC                     ;CLEAR CARRY BEFORE ROTATE
6695  026664  006100                          ROL     R0              ;SHIFT BIT POINTER
6696  026666  000640                          BR      INTG            ;BR TO SELECT NEXT DRIVE
6697  026670  013701  005530          10$:    MOV     NEWON,R1
6698                                  ;****** TO BE REMOVED *******
6699  026674  023737  005530  005526          CMP     NEWON,ONLINE
6700  026702  001002                          BNE     93$
6701  026704  000137  026304                  JMP     DRVLUP
6702  026710                          93$:
6703                                  ;       BEQ     DRVLUP
6704  026710  012703  000001                  MOV     #1,R3           ;NUMBER OF 200 MILLI-SEC STALLS
6705                                  ;LOOP1: MOV     #177777,R0
6706                                  ;LOOP2: DEC     R0
6707                                  ;       TST     R0
6708                                  ;       BNE     LOOP2
6709                                  ;       DEC     R3
6710                                  ;       BNE     LOOP1
6711                                  ;***************************
6712  026714  043701  005526                  BIC     UNLINE,R1       ;GET CHANGED ONLINE BITS
6713  026720  013737  005530  005526          MOV     NEWON,ONLINE    ;UPDATE ONLINE DRIVE BITS
6714  026726  005701                          TST     R1              ;SEE IF ANY DRIVE JUST WENT ON-LINE
6715                                  ;       BNE     97$
6716  026730  000137  026304                  JMP     DRVLUP
6717  026734                          97$:
6718                                  ;       BEQ     DRVLUP          ;BR IF NO DRIVE JUST WENT ONLINE
6719                                  ;SERVICE THE DRIVE WHICH WAS JUST SELECTED
6720  026734  012737  044156  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE USUAL  ERROR HANDLER ADDRESS
6721  026742  005037  005500                  CLR     DRIVE           ;INITIALIZE DRIVE NO.
6722  026746  010103                          MOV     R1,R3           ;COPY NEW SELECTED DRIVE BITS
6723  026750  012700  000001                  MOV     #1,R0           ;INITIALIZE BIT POINTER
6724  026754  030003                  12$:    BIT     R0,R3           ;SEE IF THIS BIT  IS  SET
6725  026756  001005                          BNE     14$             ;BR IF THIS BIT SET
6726  026760  005237  005500                  INC     DRIVE           ;INCREMENT DRIVE NO.
6727  026764  000241                          CLC                     ;CLEAR CARRY BEFORE ROTATE
6728  026766  006100                          ROL     R0              ;SHIFT BIT POINTER
6729  026770  000771                          BR      12$             ;TRY AGAIN
6730  026772  040003                  14$:    BIC     R0,R3           ;CLEAR OUT THIS BIT
6731  026774  001415                          BEQ     16$             ;BR IF ONLY ONE DRIVE SELECTED
6732                                  ;ERROR  -  MORE THAN ONE DRIVE SELECTED SIMULTANEOUSLY
6733  026776  104401  012122                  TYPE    ,MULSEL         ;REPORT ERROR
```

```
6734  027002  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;INHIBIT RK06 INTERRUPT
6735  027010  000000                          HALT                    ;HALT  FOR  MANUAL  INTERVENTION
6736                                                                  ;    PRESS 'CONT' TO  PROCEED
6737  027012  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6738  027020  004737  042510                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
6739  027024  000137  026226                  JMP     AUTAL           ;RESTART  AUTO ALIGNMENT
6740                                  ;CONTINUE WITH SELECTED DRIVE
6741  027030  020137  005524          16$:    CMP     R1,SELECT       ;SEE IF  STILL  ALIGNING SAME  DRIVE
6742  027034  001505                          BEQ     26$             ;BR IF SAME DRIVE
6743                                  ;PROCEED WITH NEWLY-SELECTED DRIVE
6744  027036  022737  000377  005534          CMP     #377,SCRACH     ;SEE IF THIS IS FIRST DRIVE SELECTED
6745  027044  001421                          BEQ     23$             ;BR IF FIRST DRIVE
6746  027046  113765  005534  000000          MOVB    SCRACH,P.DRVN(R5) ;GET LAST DRIVE NUMBER
6747  027054  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SEI READ STATUS COMMAND
6748  027062  004737  042510                  JSR     PC,DRVCAL       ;READ STATUS OF PREVIOUS DRIVE
6749  027066  032765  000040  000044          BIT     #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLOADED
6750  027074  001405                          BEQ     23$             ;BR IF HEADS NOT UNLOADED
6751  027076  112765  000111  000001          MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
6752  027104  004737  042510                  JSR     PC,DRVCAL       ;START SPINDLE ON PREVIOUS DRIVE
6753  027110  113737  005500  005534  23$:    MOVB    DRIVE,SCRACH    ;STORE DRIVE NO. FOR LATER CLEANUP STRT SPL
6754  027116  010137  005524                  MOV     R1,SELECT       ;UPDATE SELECTED DRIVE NUMBER
6755  027122  013701  005500                  MOV     DRIVE,R1        ;GET DRIVE NUMBER
6756  027126  052701  000060                  BIS     #'0,R1          ;CONVERT TO ASCII
6757  027132  110137  013007                  MOVB    R1,DRVSEL       ;GET DRIVE NUMBER INTO OUTPUT BUFFER
6758  027136  104401  013002                  TYPE    ,DRISEL         ;TYPE "DRIVE X SELECTED"
6759  027142  004737  033220                  JSR     PC,CHKDRV       ;CHECK DRIVE FOR RDY, WRITE PROT.
6760  027146  026226                          AUTAL                   ;ERROR RETURN ADDRESS
6761  027150  112737  000377  011322          MOVB    #377,UNLOD      ;SET DRIVE UNLOADED INDICATOR
6762  027156  012737  000000  005520          MOV     #0,TRACK        ;SET TO TRACK 0
6763                                  ;       MOV     #2,TRACK        ;SET TRACK = 2
6764  027164  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
6765  027172  000421                          BR      32$             ;SKIP THE REST PROGRAM FLOW
6766  027174  105737  011323          24$:    TSTB    VERIFY          ;SEE IF AUTO VERIFY
6767  027200  001016                          BNE     32$             ;SKIP UNLOAD IF VERIFY
6768  027202  112765  000107  000001          MOVB    #UNLOAD,P.CMND(R5) ;SET UNLOAD COMMAND
6769  027210  004737  042510                  JSR     PC,DRVCAL       ;UNLOAD THE DRIVE
6770  027214  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
6771  027222  004737  042510          25$:    JSR     PC,DRVCAL       ;READ STATUS OF DRIVE
6772  027226  032765  000040  000044          BIT     #S.HDHM,P.A01(R5) ;SEE IF HEADS ARE UNLODED YET
6773  027234  001772                          BEQ     25$             ;BR IF NOT YET
6774  027236  112765  000177  000001  32$:    MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6775  027244  004737  042510                  JSR     PC,DRVCAL       ;CLEAR THE S.S.
6776                                  ;       JMP     DRVLUP          ;GO BACK TO SCAN DRIVES AGAIN
6777                                  ;PROCEED WITH SAME DRIVE
6778  027250  113765  005500  000000  26$:    MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NO.
6779  027256  105137  011322                  COMB    UNLOD           ;COMPLEMENT UNLOAD INDICATOR
6780                                  ;       BNE     24$             ;BR IF DRIVE SHOULD BE UNLOADED
6781  027262  105737  011323                  TSTB    VERIFY          ;SEE IF AUTO VERIFY
6782  027266  001005                          BNE     33$             ;SKIP START SPL IF VERIFY
6783  027270  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
6784                                  ;       MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
6785  027276  004737  042510                  JSR     PC,DRVCAL       ;START SPINDLE ON THIS DRIVE
6786  027302  022737  000002  005520  33$:    CMP     #2,TRACK        ;SEE IF LAST TRACK WAS 2
6787  027310  103005                          BHIS    30$             ;NOT EXCEES HEAD 2
6788                                  ;       BNE     28$             ;BR IF IT WAS NOT 2
6789  027312  005037  005520                  CLR     TRACK           ;SET TRACK = 0
```

```
 6790  027316  000402                        BR      30$             ;GO SEEK TO ALIGNMENT CYLINDER
 6791  027320  005237  005520          28$:  INC     TRACK           ;INCREMENT TRACK NUMBER
 6792  027324  113765  005520  000005  30$:  MOVB    TRACK,P.TRCK(R5) ;SET TRACK NO.
 6793  027332  004737  033460                JSR     PC,ALNSEK       ;SEEK IN INCREMENTS TO ALIGN. CYL
 6794  027336  104401  012447                TYPE    ,HEDPOS         ;TYPE HEADS POSITIONED MSG
 6795  027342  013701  005520                MOV     TRACK,R1        ;GET CURRENT TRACK  NO.
 6796  027346  152701  000060                BISB    #'0,R1          ;CONVERT  TO  ASCII
 6797  027352  110137  012643                MOVB    R1,HEADNO       ;GET TRACK  INTO  MSG BUFFER
 6798  027356  104401  012636                TYPE    ,HEDSEL         ;TYPE "HEAD X SELECTED"
 6799  027362  005237  005520                INC     TRACK           ;INCREMENT THE TRACK NUMBER
 6800  027366  022737  000003  005520        CMP     #3,TRACK        ;ALL TRACK ARE DONE?
 6801  027374  101353                        BHI     30$             ;BRANCH IF NOT
 6802  027376  004737  030536          46$:  JSR     PC,PREPKB       ;READ THE KEYBOARD
 6803  027402  005737  005522          47$:  TST     INTCHR          ;READ ANY THING YET ?
 6804  027406  001775                        BEQ     47$             ;BRANCH IF NOT
 6805  027410  123727  005522  000032        CMPB    INTCHR,#032     ;Z ?
 6806  027416  001002                        BNE     48$             ;BRANCH IF NOT
 6807  027420  000137  026104                JMP     AUTO
 6808  027424  123727  005522  000022  48$:  CMPB    INTCHR,#022     ;R ?
 6809  027432  001002                        BNE     49$             ;
 6810  027434  000137  026104                JMP     AUTO
 6811  027440  123727  005522  000003  49$:  CMPB    INTCHR,#003     ;C ?
 6812  027446  001353                        BNE     46$             ;WAIT
 6813  027450  000137  026104                JMP     AUTO            ;TO NEXT DRIVE
 6814  027454  000137  026304                JMP     DRVLUP          ;GO BACK TO SCAN DRIVES AGAIN
 6815
 6816
 6817                                  ;AUTO RANDOM SEEK EXERCISE ROUTINE
 6818  027460                          AUTEX:
 6819  027460  004737  032250                JSR     PC,INITSS       ;INITIALIZE SUBSYSTEM
 6820  027464  004737  030536                JSR     PC,PREPKB       ;PREPARE FOR KBD INPUT
 6821  027470  104401  013025                TYPE    ,AUTEXR         ;ASK FOR SHORT OR LONG SEEK EXERCISES
 6822  027474  005737  005522          1$:   TST     INTCHR          ;SEE IF ANY INPUT
 6823  027500  001775                        BEQ     1$              ;BR IF NO INPUT
 6824  027502  023727  005522  000022        CMP     INTCHR,#022     ;SEE IF (^R) TYPED
 6825  027510  001002                        BNE     2$              ;BR IF NOT (^R)
 6826  027512  000137  024452                JMP     GIVEID          ;JUMP TO RESTART ALIGNMENT AID
 6827  027516  023727  005522  000032  2$:   CMP     INTCHR,#032     ;SEE IF (^Z) TYPED
 6828  027524  001002                        BNE     3$              ;BR IF NOT (^Z)
 6829  027526  000137  026104                JMP     AUTO            ;JUMP TO RESTART AUTO MODE
 6830  027532  023727  005522  000123  3$:   CMP     INTCHR,#'S      ;SEE IF SHORT EXERCISES DESIRED
 6831  027540  001004                        BNE     4$              ;BR IF NOT SHORT
 6832  027542  012737  002734  005534        MOV     #RNDSHT,SCRACH  ;SET 1 MINUTE SEEK COUNT
 6833  027550  000413                        BR      6$              ;PROCEED WITH SHORT EXERCISES
 6834  027552  023727  005522  000114  4$:   CMP     INTCHR,#'L      ;SEE IF LONG EXERCISES DESIRED
 6835  027560  001004                        BNE     5$              ;BR IF NOT LONG
 6836  027562  012737  016514  005534        MOV     #RNDLNG,SCRACH  ;SET 5 MINUTE SEEK COUNT
 6837  027570  000403                        BR      6$              ;PROCEED WITH LONG EXERCISES
 6838  027572  004737  030556          5$:   JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6839  027576  000730                        BR      AUTEX           ;BR TO ASK AGAIN
 6840  027600  005037  005500          6$:   CLR     DRIVE           ;SET DRIVE = 0
 6841                                  ;SELECT CURRENT DRIVE AND CHECK FOR NON-EXISTENT DRIVE INDICATION
 6842  027604  012737  030320  003046  8$:   MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
 6843  027612  113765  005500  000000        MOVB    DRIVE,P.DRVN(R5) ;SET DRIVE NUMBER
 6844  027620  013701  005500                MOV     DRIVE,R1                ;LOAD THE DRIVE NUMBER
 6845  027624  116137  005620  003124        MOVB    DTYLST(R1),TYPFMT       ;LOAD THE RK06-RK07 FLAG
```

```
 6846  027632  012737  000624  017776          MOV     #624,LCM6
 6847  027640  012737  000631  005736          MOV     #631,PRMLIM+2
 6848  027646  012737  000632  005662          MOV     #632,LC
 6849  027654  012737  000632  020000          MOV     #632,LSTCYL
 6850  027662  012737  000632  005710          MOV     #632,PRDFLT+2
 6851  027670  012737  000632  005742          MOV     #632,PRMLIM+6
 6852  027676  012737  000400  020002          MOV     #400,HOLD1
 6853  027704  012737  000025  020004          MOV     #25,HOLD3
 6854  027712  012737  010025  020006          MOV     #10025,HOLD4
 6855  027720  105737  003124                  TSTB    TYPFMT          ;IS AN RK07 ?
 6856  027724  001433                          BEQ     9$              ;BRANCH IF NOT
 6857  027726  012737  001450  017776          MOV     #1450,LCM6
 6858  027734  012737  001455  005736          MOV     #1455,PRMLIM+2
 6859  027742  012737  001456  005662          MOV     #1456,LC
 6860  027750  012737  001456  020000          MOV     #1456,LSTCYL
 6861  027756  012737  001456  005710          MOV     #1456,PRDFLT+2
 6862  027764  012737  001456  005742          MOV     #1456,PRMLIM+6
 6863  027772  012737  001000  020002          MOV     #1000,HOLD1
 6864  030000  012737  002025  020004          MOV     #2025,HOLD3
 6865  030006  012737  012025  020006          MOV     #12025,HOLD4
 6866  030014  000240                  9$:     NOP                     ;EXIT
 6867  030016  004737  032404                  JSR     PC,SCNDRV
 6868  030022  030262                          18$
 6869  030024  000240                          NOP
 6870  030026  000240                          NOP
 6871  030030  000240                          NOP
 6872  030032  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET SELECT DRIVE COMMAND
 6873  030040  012737  000377  005530          MOV     #377,NEWON      ;INITIALIZE ON-LINE INDICATOR
 6874  030046  004737  042510                  JSR     PC,DRVCAL       ;SELECT THIS DRIVE
 6875  030052  012737  044156  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
 6876  030060  022737  000377  005530          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
 6877  030066  001075                          BNE     18$             ;BR IF NED, TO SKIP THIS DRIVE
 6878                          ;CHECK DRIVE FOR RDY, WRITE PROTECT
 6879  030070  004737  033220                  JSR     PC,CHKDRV       ;CHECK THIS DRIVE
 6880  030074  027460                          AUTEX                   ;ERROR RETURN ADDRESS FOR CHKDRV
 6881                          ;PROCEED WITH SEEK EXERCISES ON THIS DRIVE
 6882  030076  013701  005500                  MOV     DRIVE,R1        ;GET DRIVE NUMBER
 6883  030102  052701  000060                  BIS     #'0,R1          ;CONVERT TO ASCII
 6884  030106  110137  013135                  MOVB    R1,AUTODR       ;GET DRIVE NUMBER INTO OUTPUT BUFFER
 6885  030112  104401  013116                  TYPE    ,AUTOEX         ;TYPE "EXERCISING DRIVE X"
 6886  030116  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 6887  030124  013700  005534                  MOV     SCRACH,R0       ;SEEK COUNT
 6888  030130  004737  030536          10$:    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 6889  030134  004737  035634                  JSR     PC,RNDADR       ;SELECT RANDOM CYLINDER ADDRESS
 6890  030140  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET RANDOM CYLINDER ADDRESS
 6891  030146  004737  042510                  JSR     PC,DRVCAL       ;DO A RANDOM SEEK
 6892  030152  005737  005522                  TST     INTCHR          ;SEE IF ANY KBD INPUT
 6893  030156  001432                          BEQ     16$             ;BR IF NO INPUT
 6894  030160  023727  005522  000022          CMP     INTCHR,#022     ;SEE IF (^R) TYPED
 6895  030166  001007                          BNE     12$             ;BR IF NOT (^R)
 6896  030170  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6897  030176  004737  042510                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6898  030202  000137  024452                  JMP     GIVEID          ;JUMP TO RESTART ALIGNMENT AID
 6899  030206  023727  005522  000032  12$:    CMP     INTCHR,#032     ;SEE IF (^Z) TYPED
 6900  030214  001007                          BNE     14$             ;BR IF NOT (^Z)
 6901  030216  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
```

```
 6902  030224  004737  042510                 JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6903  030230  000137  026104                 JMP     AUTO            ;JUMP TO RESTART AUTO MODE
 6904  030234  004737  030556          14$:   JSR     PC,ECOBAD       ;ECHO BAD INPUT
 6905  030240  004737  030536                 JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
 6906  030244  005300                 16$:   DEC     R0              ;DECREMENT SEEK COUNT
 6907  030246  001330                        BNE     10$             ;BR IF NOT DONE WITH THIS DRIVE
 6908  030250  112765  000113  000001        MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
 6909  030256  004737  042510                 JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 6910  030262  005237  005500          18$:   INC     DRIVE           ;INCREMENT DRIVE NUMBER
 6911  030266  022737  000010  005500        CMP     #10,DRIVE       ;SEE IF DONE WITH ALL DRIVES
 6912  030274  001402                        BEQ     99$
 6913  030276  000137  027604                 JMP     8$
 6914  030302                          99$:
 6915                                  ;      BNE     8$              ;BR IF MORE DRIVES TO EXERCISE YET
 6916  030302  042777  000100  150634        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 6917  030310  104401  001310                 TYPE    ,$BELL          ;RING BELL AT END OF AUTO-EXERCISES
 6918  030314  000137  026104                 JMP     AUTO            ;RESTART AUTO MODE
 6919
 6920
 6921                                  ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
 6922                                  ;  WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
 6923                                  ;    (NED = NON-EXISTENT DRIVE)
 6924  030320  032765  010000  000020  NEDHDL: BIT     #NED,P.CS2(R5)  ;SEE IF NED ON DRIVE SELECT
 6925  030326  001002                        BNE     1$              ;BR IF NED SET
 6926  030330  000137  044156                 JMP     ERRHDL          ;GO HANDLE  OTHER ERROR
 6927  030334  010046                  1$:    MOV     R0,-(SP)        ;SAVE R0,R1
 6928  030336  010146                         MOV     R1,-(SP)
 6929  030340  012700  000001                 MOV     #1,R0           ;SET BIT POINTER
 6930  030344  005001                         CLR     R1              ;CLEAR COUNTER
 6931  030346  120137  005500          2$:    CMPB    R1,DRIVE        ;SEE IF R1 = CURRENT DRIVE NUMBER
 6932  030352  001403                        BEQ     3$              ;BR IF  =
 6933  030354  005201                         INC     R1              ;INCREMENT COUNTER
 6934  030356  006300                         ASL     R0              ;SHIFT BIT POINTER
 6935  030360  000772                         BR      2$              ;TRY AGAIN
 6936  030362  040037  005530          3$:    BIC     R0,NEWON        ;CLEAR ONLINE BIT FOR  THIS DRIVE
 6937  030366  012601                         MOV     (SP)+,R1        ;RESTORE R0,R1
 6938  030370  012600                         MOV     (SP)+,R0
 6939  030372  000137  046334                 JMP     RETNML          ;TAKE  NORMAL  RETURN
 6940
 6941
 6942                                  ;;*****************************************************************
 6943                                  .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
 6944                                  ;;*****************************************************************
 6945
 6946  030376  010146                  KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
 6947  030400  017701  150542                 MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
 6948  030404  042701  177600                 BIC     #177600,R1      ;MASK OUT UNUSED BITS
 6949  030410  120127  000172                 CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
 6950  030414  003005                         BGT     20$             ;BR IF NOT
 6951  030416  120127  000141                 CMPB    R1,#141
 6952  030422  002402                         BLT     20$             ;BR IF NOT
 6953  030424  042701  000040                 BIC     #BIT5,R1        ;MAKE IT UPPER CASE
 6954  030430  010137  005522          20$:   MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
 6955  030434  122701  000003                 CMPB    #003,R1         ;SEE IF (^C) TYPED
 6956  030440  001007                         BNE     2$              ;BR IF NOT (^C)
 6957  030442  104401  013205                 TYPE    ,CNTRLC         ;ECHO (^C)
```

```
 6958  030446  042777  000100  150470  1$:     BIC     #BIT6,@$TKS     ;CLEAR KBD INTERRUPT ENABLE BIT
 6959  030454  012601                          MOV     (SP)+,R1        ;RESTORE R1
 6960  030456  000002                          RTI                     ;RETURN
 6961  030460  122701  000032          2$:     CMPB    #032,R1         ;SEE IF (^Z) TYPED
 6962  030464  001003                          BNE     3$              ;BR IF NOT (^Z)
 6963  030466  104401  013212                  TYPE    ,CNTRLZ         ;ECHO (^Z)
 6964  030472  000765                          BR      1$              ;RETURN
 6965  030474  122701  000022          3$:     CMPB    #022,R1         ;SEE IF (^R) TYPED
 6966  030500  001003                          BNE     4$              ;BR IF NOT (^R)
 6967  030502  104401  013217                  TYPE    ,CNTRLR         ;ECHO (^R)
 6968  030506  000757                          BR      1$              ;RETURN
 6969  030510  122701  000007          4$:     CMPB    #007,R1         ;SEE IF (^G) TYPED
 6970  030514  001003                          BNE     5$              ;BR IF NOT (^G)
 6971  030516  104401  013231                  TYPE    ,CNTRLG         ;ECHO (^G)
 6972  030522  000751                          BR      1$              ;RETURN
 6973  030524  104401  005522          5$:     TYPE    ,INTCHR         ;ECHO INPUT
 6974  030530  104401  001315                  TYPE    ,$CRLF          ;DO <CR> AND <LF>
 6975  030534  000744                          BR      1$              ;RETURN
 6976
 6977
 6978
 6979                                  ;;*****************************************************************
 6980                                  .SBTTL  PREPKB - PREPARE FOR KEYBOARD INPUT
 6981                                  ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
 6982                                  ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
 6983                                  ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
 6984                                  ;*ENABLES KBD INTERRUPT.
 6985                                  ;* CALL:
 6986                                  ;*      JSR     PC,PREPKB
 6987                                  ;;*****************************************************************
 6988
 6989  030536  005077  150404          PREPKB: CLR     @$TKB           ;CLEAR KBD BUFFER AND DONE BIT
 6990  030542  005037  005522                  CLR     INTCHR          ;CLEAR TTY INPUT WORD
 6991  030546  052777  000100  150370          BIS     #BIT6,@$TKS     ;ENABLE KBD INTERRUPT
 6992  030554  000207                          RTS     PC              ;RETURN
 6993
 6994
 6995
 6996                                  ;;*****************************************************************
 6997                                  .SBTTL  ECOBAD - ECHO BAD TTY INPUT
 6998                                  ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
 6999                                  ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
 7000                                  ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
 7001                                  ;*AND <LF> ARE DONE.
 7002                                  ;* CALL -     JSR      PC,ECOBAD
 7003                                  ;;*****************************************************************
 7004
 7005  030556  104401  005522          ECOBAD: TYPE    ,INTCHR         ;ECHO BAD CHARACTER
 7006  030562  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>, <LF>
 7007  030566  000207                          RTS     PC              ;RETURN
 7008
 7009
 7010
 7011                                  ;;*****************************************************************
 7012                                  ;*SIZMEM - SIZE MEMORY, SET LIMITS
 7013                                  ;*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).
                                        ;;*****************************************************************
030570  104407                  SIZMEM: SAVREG                  ;SAVE R0-R5
030572  012737  000200  057710          MOV     #200,$KT11      ;SET MEM MGT KEY FOR $SIZE
030600  004737  057652                  JSR     PC,$SIZE        ;SIZE MEMORY
030604  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
030610  100406                          BM      8$              ;BR IF MEM MGT PRESENT
030612  013737  060154  005574          MOV     $LSTAD,MAHILM   ;SET MEM LIMIT LO BITS
030620  005037  005576                  CLR     MAHILM+2        ;CLEAR MEM LIMIT HI BITS
030624  000436                          BR      16$             ;GO TYPE LAST ADDRESS
                                ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
030626  013700  060156          8$:     MOV     $LSTBK,R0       ;LO BITS
030632  005001                          CLR     R1              ;HI BITS
030634  012702  000006                  MOV     #6,R2           ;SET LOOP COUNT = 6
030640  000241                  12$:    CLC                     ;ROTATE LOOP
030642  006100                          ROL     R0
030644  006101                          ROL     R1
030646  005302                          DEC     R2
030650  001373                          BNE     12$
                                ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
030652  063700  060154                  ADD     $LSTAD,R0       ;ADD LO BITS
030656  005501                          ADC     R1              ;HI BITS
030660  010037  005574                  MOV     R0,MAHILM       ;SET LO BITS OF MEM LIMIT
030664  010137  005576                  MOV     R1,MAHILM+2     ;SET HI BITS OF MEM LIMIT
030670  000241                          CLC
030672  006100                          ROL     R0
030674  006101                          ROL     R1
030676  006100                          ROL     R0
030700  006101                          ROL     R1
030702  006100                          ROL     R0
030704  006101                          ROL     R1
030706  020127  000037                  CMP     R1,#37
030712  103403                          BLO     16$
030714  112737  000001  003143          MOVB    #1,UBMPRS       ;SET "UNIBUS MAP PRESENT" FLAG
                                ;TYPE "LAST PHYS MEM ADR   XXXXXXXX"
030722  104401  007101          16$:    TYPE    ,LSTMEM         ;TYPE "LAST PHYS MEM ADR ="
030726  012746  005574                  MOV     #MAHILM,-(SP)   ;PUT POINTER ON STACK
030732  004737  055414                  JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL ASCII
030736  004737  055730                  JSR     PC,@#$SUPRS     ;TYPE "XXXXXXXX"
030742  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
030746  104401  001315                  TYPE    ,$CRLF
030752  104410                          RESREG                  ;RESTORE R0-R5
030754  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        ;* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
                                        ;*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
                                        ;*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
                                        ;*TOP OF STACK.
                                        ;;*****************************************************************
030756  016646  000002          GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
030762  104403                          TYPOS                   ;TYPE IT
```

```
7070  030764  006                          .BYTE   6               ;SIX DIGITS
7071  030765  000                          .BYTE   0               ;SUPPRESS LEADING ZEROS
7072  030766  104401  007434               TYPE    ,NEWMSG         ;TYPE    "  NEW = "
7073  030772  004737  034002               JSR     PC,RDCHRS       ;READ NEW VALUE FROM KBD
7074  030776  031030                       7$                      ;(^C) RETURN ADDRESS
7075  031000  031030                       7$                      ;(^Z) RETURN ADDRESS
7076  031002  031030                       7$                      ;(^U) OR ERROR RETURN ADDRESS
7077  031004  005700                       TST     R0              ;SEE IF ANY CHARS TYPED
7078  031006  001001                       BNE     4$              ;BR IF YES
7079  031010  000207                       RTS     PC              ;RETURN - OLD VALUE UNCHANGED
7080  031012  020027  000006       4$:     CMP     R0,#6           ;SEE IF > 6 CHARS TYPED
7081  031016  003407                       BLE     8$              ;BR IF NOT BAD
7082  031020  104401  005264       6$:     TYPE    ,BUFF0          ;ECHO BAD INPUT
7083  031024  104401  001314               TYPE    ,$QUES
7084  031030  162716  000010       7$:     SUB     #10,(SP)        ;FIX ERROR RETURN PC
7085  031034  000207                       RTS     PC              ;ERROR RETURN
7086  031036  012746  005264       8$:     MOV     #BUFF0,-(SP)    ;PUT POINTER TO CHARS ON STACK
7087  031042  004737  054456               JSR     PC,OCTBIN       ;CONVERT DIGITS TO BINARY
7088  031046  031020                       6$                      ;ERROR RETURN ADDRESS
7089  031050  012600                       MOV     (SP)+,R0        ;GET NEW BINARY VALUE
7090  031052  005737  054610               TST     $HIOCT          ;SEE IF HI BITS ARE 0
7091  031056  001360                       BNE     6$              ;BR IF NOT
7092  031060  010066  000002               MOV     R0,2(SP)        ;PUT NEW VALUE ON STACK
7093  031064  000207                       RTS     PC              ;RETURN
7094
7095
7096
7097                                   ;;*****************************************************************
7098                                   .SBTTL  GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION
7099                                   ;*THIS SUBROUTINE ALLOWS THE CONTENTS OF THE SOFTWARE SWITCH
7100                                   ;*REGISTER TO BE MODIFIED BY TTY INPUT. THE SUBROUTINE TYPES
7101                                   ;*  "SWR = XXXXXX  NEW = ", AND WAITS FOR A NEW OCTAL VALUE
7102                                   ;*OF UP TO SIX DIGITS TO BE TYPED.
7103                                   ;;*****************************************************************
7104  031066  022737  000176  001140  GTSWRG: CMP     #SWREG,SWR      ;SEE IF SOFTWARE SWR SELECTED
7105  031074  001010                          BNE     6$              ;BR IF NOT
7106  031076  013746  000176                  MOV     SWREG,-(SP)     ;PUT OLD VALUE ON STACK
7107  031102  104401  007425                  TYPE    ,SWRMSG         ;TYPE "SWR = "
7108  031106  004737  030756                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
7109  031112  012637  000176                  MOV     (SP)+,SWREG     ;STORE NEW VALUE
7110  031116  000207                  6$:     RTS     PC              ;RETURN
7111
7112
7113
7114                                   ;;*****************************************************************
7115                                   ;*INITMM - INITIALIZE MEMORY MANAGEMENT REGISTERS
7116                                   ;*THIS SUBROUTINE INITIALIZES KIPAR'S AND KIPDR'S, FOR
7117                                   ;*CONTIGUOUS 4K PAGES, STARTING AT PHYSICAL ADDRESS 0.
7118                                   ;*KIPAR7 IS LOADED WITH 177600, TO PRESERVE THE I/O PAGE.
7119                                   ;*22-BIT MODE AND THE UNIBUS MAP ARE ENABLED, IF PRESENT,
7120                                   ;*BUT MEM MGT IS NOT TURNED ON IN THIS SUBROUTINE. HOWEVER,
7121                                   ;*MEM MGT TRAPS ARE ENABLED.
7122
7123                                   ;;*****************************************************************
7124  031120  104407                  INITMM: SAVREG                  ;SAVE R0-R5
7125  031122  005001                          CLR     R1              ;INIT FOR PAR LOADING
```

```
 7126  031124  012702  172340                MOV     #KIPAR0,R2      ;ADDR OF FIRST PAR
 7127  031130  012703  000010                MOV     #8.,R3          ;LOAD 8 PAR'S AND 8 PDR'S
 7128  031134  012762  077406  177740  4$:   MOV     #77406,-40(R2)  ;PDR = 4K,UP,READ/WRITE
 7129  031142  010122                        MOV     R1,(R2)+        ;LOAD A PAR
 7130  031144  062701  000200                ADD     #200,R1         ;UPDATE FOR NEXT PAR
 7131  031150  005303                        DEC     R3              ;DECREMENT LOOP COUNTER
 7132  031152  001370                        BNE     4$              ;LOOP UNTIL ALL 8 ARE LOADED
 7133  031154  012742  177600                MOV     #177600,-(R2)   ;SET UP KIPAR7 FOR I/O PAGE
 7134  031160  105737  003143                TSTB    UBMPRS          ;SEE IF 22-BIT ADDRESSES
 7135  031164  001403                        BEQ     10$             ;BR IF NOT
 7136  031166  012737  000060  172516        MOV     #60,@#SR3       ;ENABLE 22-BIT MODE,AND UNIBUS MAP
 7137  031174  012737  001000  177572  10$:   MOV     #BIT9,@#SR0     ;ENABLE KT11 TRAPS
 7138  031202  104410                        RESREG                  ;RESTORE R0-R5
 7139  031204  000207                        RTS     PC              ;RETURN
 7140
 7141
 7142
 7143                                ;;**********************************************************************
 7144                                ;*SERVICE ROUTINE FOR MEM MGT TRAPS
 7145                                ;;**********************************************************************
```

```
7146  031206  004737  043676          KTERHD: JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
7147  031212  013737  177572  001174          MOV     @#SR0,$REG5     ;GET SR0
7148  031220  013737  177574  001176          MOV     @#SR1,$REG6     ;GET SR1
7149  031226  013737  177576  001200          MOV     @#SR2,$REG7     ;GET SR2
7150  031234  012737  031260  000004          MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
7151  031242  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
7152  031250  013737  172516  001202          MOV     @#SR3,$REG10    ;GET SR3
7153  031256  000406                          BR      10$
7154  031260  022626                  8$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
7155  031262  112737  000003  065610          MOVB    #3,DF30+18.     ;FIX PRINTOUT FOR NO SR3
7156  031270  105037  064250                  CLRB    DH704+11.
7157  031274  104121                  10$:    ERROR   121             ;KT11 FAILURE
7158  031276  000137  046126                  JMP     HLTPRG          ;ABORT !!!
7159
7160
7161
7162                                    ;;*******************************************************************
7163                                    ;*ENBCSR - ENABLE MEMORY CSR'S FOR PARITY ERRORS
7164                                    ;;*******************************************************************
7165  031302  010146                  ENBCSR: MOV     R1,-(SP)        ;SAVE R1
7166  031304  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
7167  031310  013746  000006                  MOV     @#ERRVEC+2,-(SP)
7168  031314  012737  031336  000004          MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
7169  031322  012701  172100                  MOV     #MEMCSR,R1      ;ADRS OF MEMORY CSR'S
7170  031326  005011                  4$:     CLR     (R1)            ;ZERO THIS CSR
7171  031330  012711  000001                  MOV     #BIT0,(R1)      ;SET ENABLE IN THIS CSR
7172  031334  000401                          BR      10$
7173  031336  022626                  8$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
7174  031340  062701  000002          10$:    ADD     #2,R1           ;INCREMENT TO NEX. CSR ADRS
7175  031344  020127  172140                  CMP     R1,#MEMCSR+40   ;SEE IF DONE CHECKING YET
7176  031350  001366                          BNE     4$              ;BR IF NOT
7177  031352  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;RESTORE OLD VECTORS
7178  031356  012637  000004                  MOV     (SP)+,@#ERRVEC
7179  031362  012601                          MOV     (SP)+,R1        ;RESTORE R1
7180  031364  000207                          RTS     PC              ;RETURN
7181
7182
7183
7184                                    ;;*******************************************************************
7185                                    ;*SERVICE ROUTINE FOR MEM PARITY ERRORS
7186                                    ;;*******************************************************************
7187  031366  010146                  MPERHD: MOV     R1,-(SP)        ;SAVE R1
7188  031370  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
7189  031374  013746  000006                  MOV     @#ERRVEC+2,-(SP)
7190  031400  004737  043676                  JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
7191  031404  016637  000006  001174          MOV     6(SP),$REG5     ;GET PC OF ERROR
7192  031412  012737  031536  000004          MOV     #10$,@#ERRVEC   ;SET T.O. VECTOR
7193  031420  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
7194                                  ;HANDLE 11/70 MEMORY PARITY ERROR
7195  031426  013737  177740  001176          MOV     @#LOERAD,$REG6  ;LOW ERROR ADRS REG
7196  031434  013737  177742  001200          MOV     @#HIERAD,$REG7  ;HI ERROR ADRS REG
7197  031442  013737  177744  001202          MOV     @#MEMSYS,$REG10 ;MEMORY SYSTEM REG
7198  031450  012737  064304  065632          MOV     #DH707,DF31+16. ;FIX ERROR MSG FOR 11/70
7199  031456  112737  000004  065634          MOVB    #4,DF31+18.
7200  031464  104122                          ERROR   122             ;11/70 MEM PARITY ERROR
7201  031466  005737  001200                  TST     $REG7           ;SEE IF BAD MEMORY IS IN PROGRAM AREA
```

```
 7202  031472  001011                          BNE     6$                      ;BR IF NOT
 7203  031474  023727  001176  073636          CMP     $REG6,#RWBUF+6000 ;SEE IF BAD MEM IS IN PROG. AREA
 7204  031502  103005                          BHIS    6$                      ;BR IF NOT
 7205  031504  105737  003144          4$:     TSTB    MEMABT                  ;SEE IF ABORT DESIRED
 7206  031510  001002                          BNE     6$                      ;BR IF NOT
 7207  031512  000137  046126                  JMP     HLTPRG                  ;ABORT !!!
 7208  031516  012637  000006          6$:     MOV     (SP)+,@#ERRVEC+2 ;RESTORE T.O. VECTOR
 7209  031522  012637  000004                  MOV     (SP)+,@#ERRVEC
 7210  031526  012601                          MOV     (SP)+,R1                ;RESTORE R1
 7211  031530  004737  031302                  JSR     PC,ENBCSR               ;GO CLEAR AND ENABLE CSR'S
 7212  031534  000002                          RTI                             ;RETURN
 7213                                  ;HANDLE ALL OTHER MEMORY PARITY ERRORS  (NON-11/70)
 7214  031536  022626                  10$:    CMP     (SP)+,(SP)+             ;CLEAN UP THE STACK
 7215  031540  012737  031570  000004          MOV     #14$,@#ERRVEC           ;SET T.O. VECTOR
 7216  031546  012701  172100                  MOV     #MEMCSR,R1              ;GET FIRST CSR ADDRESS
 7217  031552  011137  001200          12$:    MOV     (R1),$REG7              ;CHECK FOR A MEMORY CSR
 7218  031556  100005                          BPL     16$                     ;BR IF NO ERROR SET HERE
 7219  031560  010137  001176                  MOV     R1,$REG6                ;GET CSR ADDRESS
 7220  031564  104122                          ERROR   122                     ;MEMORY PARITY ERROR (NON-11/70)
 7221  031566  000746                          BR      4$                      ;GO SEE IF SHOULD ABORT
 7222  031570  022626                  14$:    CMP     (SP)+,(SP)+             ;CLEAN UP STACK
 7223  031572  062701  000002          16$:    ADD     #2,R1                   ;INCR R1 TO POINT TO NEXT CSR
 7224  031576  020127  172140                  CMP     R1,#MEMCSR+40           ;SEE IF DONE CHECKING
 7225  031602  103763                          BLO     12$                     ;BR IF NOT
 7226  031604  000744                          BR      6$                      ;RETURN
 7227
 7228
 7229
 7230                                  ;;*****************************************************************
 7231                                  ;*PREPAR - PREPARE MEM MGT FOR RELOCATION
 7232                                  ;*THIS SUBROUTINE CALLS INITMM, AND THEN SETS UP CONSTANT
 7233                                  ;*FOR KIPAR6 IN SAVPAR, AND LOADS THE UNIBUS MAP REGISTERS,
 7234                                  ;*IF PRESENT.
 7235                                  ;;*****************************************************************
 7236  031606  104407                  PREPAR: SAVREG                          ;SAVE R0-R5
 7237  031610  004737  031120                  JSR     PC,INITMM               ;INIT MEM MGT REGISTERS
 7238  031614  013700  005604                  MOV     PMA,R0                  ;LO BITS OF MA
 7239  031620  042700  017777                  BIC     #17777,R0               ;MASK FOR BITS 13-15
 7240  031624  013701  005606                  MOV     PMA+2,R1                ;HI BITS OF MA
 7241  031630  010003                          MOV     R0,R3                   ;SAVE THESE BITS
 7242  031632  010104                          MOV     R1,R4
 7243  031634  006100                          ROL     R0                      ;GET MA BITS 13-21 INTO R1 BITS 7-15
 7244  031636  006101                          ROL     R1
 7245  031640  006100                          ROL     R0
 7246  031642  006101                          ROL     R1
 7247  031644  000301                          SWAB    R1
 7248  031646  006100                          ROL     R0
 7249  031650  106001                          RORB    R1
 7250  031652  010137  003176                  MOV     R1,SAVPAR               ;CONSTANT FOR LOADING PAR6 LATER
 7251  031656  105737  003143                  TSTB    UBMPRS                  ;SEE IF UNIBUS MAP ENABLED
 7252  031662  001420                          BEQ     9$                      ;BR IF NOT (NO UNIBUS MAPPING)
 7253                                  ;PREPARE FOR 22-BIT ADDRESSING -  LOAD UNIBUS MAP REGISTERS
 7254  031664  012700  170200                  MOV     #MAPL00,R0              ;STARTING ADDR OF MAP REGISTERS
 7255  031670  012701  000037                  MOV     #31.,R1                 ;SET REGISTER COUNTER
 7256  031674  010320                  4$:     MOV     R3,(R0)+                ;LOAD A MAP REGISTER
 7257  031676  010420                          MOV     R4,(R0)+
```

```
7258  031700  062703  020000              ADD     #20000,R3       ;ADD 4K WORDS
7259  031704  005504                      ADC     R4
7260  031706  077106                      SOB     R1,4$           ;LOOP UNTIL 31(DEC) ARE LOADED
7261  031710  042765  160000  000010      BIC     #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
7262  031716  142765  000003  000007      BICB    #3,P.BAHI(R5)
7263  031724  104410              9$:     RESREG                  ;RESTORE R0-R5
7264  031726  000207                      RTS     PC              ;RETURN
7265
7266
7267
7268                                      ;;*********************************************************************
7269                                      ;* FNDXDP - FIND STARTING ADR OF XXDP LOADER , AND STORE IT
7270                                      ;*IN XXDPAD.
7271                                      ;;*********************************************************************
7272  031730  005737  005576      FNDXDP: TST     MAHILM+2        ;TEST HI BITS OF UPPER MEMORY LIMIT
7273  031734  001404                      BEQ     6$              ;BR IF HI BITS ARE 0
7274  031736  012737  160000  005540  4$: MOV     #160000,XXDPAD  ;START OF 28K IS END OF XXDP
7275  031744  000412                      BR      8$
7276  031746  023727  005574  157776  6$: CMP     MAHILM,#157776  ;SEE IF MEM LIMIT > OR = 157776
7277  031754  103370                      BHIS    4$              ;BR IF YES
7278  031756  013737  005574  005540      MOV     MAHILM,XXDPAD   ;HI MEM LIMIT IS END OF XXDP
7279  031764  062737  000002  005540      ADD     #2,XXDPAD       ;ADD 2 BYTES
7280  031772  162737  006000  005540  8$: SUB     #6000,XXDPAD    ;COMPUTE START OF XXDP
7281  032000  000207                      RTS     PC              ;RETURN
7282
7283
7284
7285                                      ;;*********************************************************************
7286                                      ;* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
7287                                      ;*THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
7288                                      ;*PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
7289                                      ;*STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
7290                                      ;*
7291                                      ;* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
7292                                      ;*THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
7293                                      ;*STORED IN XDPSAV,XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
7294                                      ;*ADDRESS).
7295                                      ;;*********************************************************************
7296  032002  104407              SAVXDP: SAVREG                  ;SAVE R0-R5
7297  032004  005004                      CLR     R4              ;SET INDICATOR TO SAVE XXDP
7298  032006  000410                      BR      XDP1
7299  032010  104407              GETXDP: SAVREG                  ;SAVE R0-R5
7300  032012  105037  003133              CLRB    XOVLAD          ;CLEAR XXDP OVERLAID INDICATOR
7301  032016  105737  003134              TSTB    XDPSVD          ;SEE IF XXDP WAS SAVED
7302  032022  001425                      BEQ     XDP2            ;BR IF NOT SAVED
7303  032024  012704  000001              MOV     #1,R4           ;SET INDICATOR TO GET XXDP
7304  032030  005737  000170      XDP1:   TST     KILLDR          ;SEE IF OK TO OVERLAY LOADER
7305  032034  001020                      BNE     XDP2            ;BR IF YES, TO EXIT
7306  032036  012702  003000              MOV     #1536.,R2       ;GET SET TO MOVE 1536(DEC) WORDS
7307  032042  013703  005540              MOV     XXDPAD,R3       ;GET ORIG ADR OF START OF XXDP
7308  032046  013700  005542              MOV     XDPSAV,R0       ;GET SAVE ADR LO BITS
7309  032052  013701  005544              MOV     XDPSAV+2,R1     ;HI BITS
7310  032056  005737  057710              TST     $KT11           ;SEE IF MEM MGT PRESENT
7311  032062  100417                      BMI     XDP3            ;BR IF PRESENT
7312  032064  005704                      TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
7313  032066  001005                      BNE     XDP4            ;BR IF WANT TO GET XXDP
```

```
 7314  032070  012320                  6$:     MOV     (R3)+,(R0)+     ;SAVE A WORD
 7315  032072  005302                          DEC     R2              ;SEE IF 1536(DEC) WORDS YET
 7316  032074  001375                          BNE     6$              ;BR IF NOT YET
 7317  032076  104410                  XDP2:   RESREG                  ;RESTORE R0-R5
 7318  032100  000207                          RTS     PC              ;RETURN
 7319  032102  062700  006000          XDP4:   ADD     #6000,R0        ;POINT TO END OF SAVE AREA
 7320  032106  062703  006000                  ADD     #6000,R3        ;POINT TO END OF XXDP LOADER AREA
 7321  032112  014043                  8$:     MOV     -(R0),-(R3)     ;GET A WORD
 7322  032114  005302                          DEC     R2              ;SEE IF 1536(DEC) WORDS YET
 7323  032116  001375                          BNE     8$              ;BR IF NOT YET
 7324  032120  000766                          BR      XDP2            ;GO EXIT
 7325                                  ;COME HERE IF MEM MGT
 7326  032122  004737  031120          XDP3:   JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
 7327  032126  006100                          ROL     R0              ;GET ADR BITS 13-21 INTO R1 BITS 7-15
 7328  032130  006101                          ROL     R1
 7329  032132  006100                          ROL     R0
 7330  032134  006101                          ROL     R1
 7331  032136  000301                          SWAB    R1
 7332  032140  006100                          ROL     R0
 7333  032142  106001                          RORB    R1
 7334  032144  010137  172354                  MOV     R1,@#KIPAR6     ;SET UP PAR6
 7335  032150  013701  005542                  MOV     XDPSAV,R1       ;GET LO ADRS BITS
 7336  032154  042701  160000                  BIC     #160000,R1      ;FIX UP R1 TO REFERENCE PAR6
 7337  032160  052701  140000                  BIS     #140000,R1
 7338  032164  005704                  16$:    TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
 7339  032166  001007                          BNE     18$             ;BR IF WANT TO GET XXDP
 7340  032170  012305                          MOV     (R3)+,R5        ;SAVE A WORD
 7341  032172  005237  177572                  INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
 7342  032176  010521                          MOV     R5,(R1)+
 7343  032200  005337  177572                  DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
 7344  032204  000406                          BR      20$
 7345  032206  005237  177572          18$:    INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
 7346  032212  012105                          MOV     (R1)+,R5        ;GET A WORD
 7347  032214  005337  177572                  DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
 7348  032220  010523                          MOV     R5,(R3)+
 7349  032222  032701  020000          20$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO PAGE 7
 7350  032226  001405                          BEQ     22$             ;BR IF NOT
 7351  032230  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 7352  032234  062737  000200  172354          ADD     #200,@#KIPAR6   ;UPDATE PAR BY 4K
 7353  032242  005302                  22$:    DEC     R2              ;SEE IF 1536 WORDS YET
 7354  032244  001347                          BNE     16$             ;BR IF NOT YET
 7355  032246  000713                          BR      XDP2            ;BR TO RETURN
 7356
 7357
 7358
 7359                                  ;;*****************************************************************
 7360                                  .SBTTL  INITSS - INITIALIZE SUBSYSTEM
 7361                                  ;*
 7362                                  ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
 7363                                  ;*AND DOES A SUBSYSTEM CLEAR.
 7364                                  ;* USES -   R2,R5
 7365                                  ;* CALL:
 7366                                  ;*              JSR     PC,INITSS
 7367                                  ;;*****************************************************************
 7368
 7369  032250  012737  044156  003046  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
```

```
7370  032256  012737  042700  003044          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
7371  032264  013702  003036                  MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
7372  032270  012705  002630                  MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
7373  032274  105037  003142                  CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
7374  032300  004737  032356                  JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
7375  032304  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
7376  032312  004737  042510                  JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
7377  032316  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
7378  032324  113737  005500  002714          MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
7379  032332  113765  003125  000007          MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
7380  032340  153765  003124  000007          BISB    TYPFMT,P.CS1H(R5)       ;SET DRV TYP
7381  032346  116537  000007  002723          MOVB    P.CS1H(R5),P.CS1H+PARM1  ;COPY FORMAT & DR TYPE TO PARM1
7382  032354  000207                          RTS     PC              ;SUBROUTINE EXIT
7383
7384
7385
7386                                  ;;*****************************************************************
7387                                  .SBTTL  CLRPRM - CLEAR DRIVER INPUT PARAMETERS
7388                                  ;*THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
7389                                  ;*PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
7390                                  ;* CALL -       JSR     PC,CLRPRM
7391                                  ;;*****************************************************************
7392
7393  032356  010046                  CLRPRM: MOV     R0,-(SP)        ;SAVE R0
7394  032360  010546                          MOV     R5,-(SP)        ;SAVE R5
7395  032362  010500                          MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
7396  032364  062705  000016                  ADD     #P.CS1,R5       ;COMPUTE LIMIT ADDRESS
7397  032370  005020                  1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
7398  032372  020005                          CMP     R0,R5           ;SEE IF DONE YET
7399  032374  001375                          BNE     1$              ;BR IF  NOT DONE  YET
7400  032376  012605                          MOV     (SP)+,R5        ;RESTORE R5
7401  032400  012600                          MOV     (SP)+,R0        ;RESTORE R0
7402  032402  000207                          RTS     PC              ;RETURN
7403
7404
7405
7406                                  ;;*****************************************************************
7407                                  .SBTTL  SCNDRV - SCAN DRIVE FOR STATUS
7408                                  ;*THIS SUBROUTINE SCANS THE SELECTED DRIVE TO INSURE
7409                                  ;*THAT THE DRIVE IS ON-LINE, READY, NOT WRITE-LOCKED,
7410                                  ;*AND NOT LOADED WITH AN ALIGNMENT CARTRIDGE. IF ANY
7411                                  ;*OF THESE CONDITIONS ,RE NOT MET, AN APPROPRIATE
7412                                  ;*MESSAGE IS TYPED, RK06 INTERRUPT IS DISABLED, AND
7413                                  ;*A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7414                                  ;*AFTER THE CALL TO SCNDRV. THE REQUESTED DRIVE NO.
7415                                  ;*MUST BE PASSED TO THE SUBROUTINE IN WORD "DRIVE".
7416                                  ;*ALSO, IF RUNNING FROM ADRS 200 AND RK06 IS LOAD MEDIUM,
7417                                  ;*DRIVE 0 WILL BE REJECTED FOR USE.
7418                                  ;* CALL -    JSR      PC,SCNDRV
7419                                  ;*           <ERROR RETURN ADDRESS>
7420                                  ;*
7421                                  ;;*****************************************************************
7422
7423  032404  105037  003124          SCNDRV: CLRB    TYPFMT          ;ASSUME RK06 TO START
7424  032410  004737  032250                  JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
7425  032414  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;GET DRIVE NUMBER
```

```
7426  032422  042712  000100                  BIC     #IE,(R2)          ;DISABLE RK06 INTERRUPT
7427  032426  113762  005500  000010          MOVB    DRIVE,RKCS2(R2)   ;SET DRIVE NO. IN RKCS2
7428  032434  105737  003124                  TSTB    TYPFMT            ;SEE IF RK07
7429  032440  001003                          BNE     5$                ;BR IF YES
7430  032442  012712  000001                  MOV     #GO,(R2)          ;ELSE SET GO  BIT IN CS1 FOR RK06
7431  032446  000402                          BR      7$
7432  032450  012712  002001          5$:     MOV     #<CDT!GO>,(R2)
7433  032454  005037  005534          7$:     CLR     SCRACH            ;CLEAR STALL COUNTER
7434  032460  005237  005534          14$:    INC     SCRACH            ;INCREMENT STALL COUNTER
7435  032464  001375                          BNE     14$               ;STALL FOR SEVERAL MILLI-SEC
7436  032466  032762  001000  000010          BIT     #MDS,RKCS2(R2)    ;SEE IF MDS ERROR
7437  032474  001417                          BEQ     18$               ;BR IF NOT
7438  032476  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET COMMAND FOR ERROR REPORT
7439  032504  011265  000016                  MOV     (R2),P.CS1(R5)    ;GET RKCS1
7440  032510  004737  052204                  JSR     PC,I.CST1         ;GET OTHER RK611 REGS FOR REPORT
7441  032514  004737  043676                  JSR     PC,REPSUP         ;SET UP ERROR REPORT
7442  032520  104052                          ERROR   52                ;REPORT MDS ERROR
7443  032522  052737  000200  005474          BIS     #ABORT,RECODE     ;SET ABORT FLAG
7444  032530  000137  045752                  JMP     ALLTRM            ;GO ABORT TESTING
7445  032534  032762  000040  000014  18$:     BIT     #DTYE,RKER(R2)    ;SEE IF DRV TYP ERR
7446  032542  001403                          BEQ     20$               ;BR IF NO=RK06
7447  032544  152737  000004  003124          BISB    #B.CDT,TYPFMT     ;ELSE SET FOR RK07
7448  032552  004737  032250          20$:    JSR     PC,INITSS         ;INIT THE SUBSYSTEM
7449  032556  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
7450  032564  012737  030320  003046          MOV     #NEDHDL,A.ABNL    ;SET NED ABNORMAL RETURN ADDRESS
7451  032572  012737  000377  005530          MOV     #377,NEWON        ;INIT. ON-LINE INDICATOR
7452  032600  004737  042510                  JSR     PC,DRVCAL         ;READ ALL DRIVE STATUS
7453                          ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
7454  032604  012737  044156  003046          MOV     #ERRHDL,A.ABNL    ;RESTORE ABNL RETURN ADDRESS
7455  032612  022737  000377  005530          CMP     #377,NEWON        ;SEE IF NED INDICATION ON THIS DRIVE
7456  032620  001430                          BEQ     4$                ;BR IF NED NOT SET
7457  032622  005737  003150                  TST     AUTOFG            ;FORM AUTO MODE ?
7458  032626  001017                          BNE     2$
7459  032630  113737  005500  007467          MOVB    DRIVE,BADDRV+6    ;GET DRIVE NO. INTO BUF
7460  032636  152737  000060  007467          BISB    #'0,BADDRV+6      ;CONVERT TO ASCII
7461  032644  104401  007461                  TYPE    ,BADDRV           ;TYPE "DRIVE X"
7462  032650  104401  007472                  TYPE    ,NXDRIV           ;TYPE "NON-EXISTENT"
7463  032654  132737  000200  001341          BITB    #BIT7,$ENVM       ;SEE IF APT
7464  032662  001401                          BEQ     2$                ;BR IF NO
7465  032664  104123                          ERROR   123               ;NED UNDER APT SIZING
7466                          ;SERVICE ERRORS HERE
7467  032666  042762  000100  000000  2$:     BIC     #IE,RKCS1(R2)     ;DISABLE RK06 INTERRUPT
7468  032674  017616  000000                  MOV     @(SP),(SP)        ;SET UP ERROR RETURN ADDRESS
7469  032700  000207                          RTS     PC                ;ERROR RETURN
7470                          ;SEE IF DRIVE IS READY
7471  032702  032765  000200  000040  4$:     BIT     #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
7472  032710  001016                          BNE     6$                ;BR IF DRIVE IS READY
7473  032712  005737  003150                  TST     AUTOFG
7474  032716  001363                          BNE     2$                ;SKIP IF FROM AUTO MODE
7475  032720  113737  005500  007467          MOVB    DRIVE,BADDRV+6    ;GET DRIVE NO. INTO BUF
7476  032726  152737  000060  007467          BISB    #'0,BADDRV+6      ;CONVERT TO ASCII
7477  032734  104401  007461                  TYPE    ,BADDRV           ;TYPE "DRIVE X"
7478  032740  104401  007506                  TYPE    ,NTREDY           ;TYPE "NOT READY"
7479  032744  000750                          BR      2$                ;TAKE ERROR EXIT
7480                          ;SEE IF DRIVE IS WRITE ENABLED
7481  032746  032765  004000  000040  6$:     BIT     #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
```

```
7482  032754  001416                        BEQ     8$              ;BR IF WRITE LOCK NOT SET
7483  032756  005737  003150                TST     AUTOFG
7484  032762  001013                        BNE     8$              ;FROM THE AUTO MODE
7485  032764  113737  005500  007467        MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
7486  032772  152737  000060  007467        BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
7487  033000  104401  007461                TYPE    ,BADDRV         ;TYPE "DRIVE X"
7488  033004  104401  007520                TYPE    ,WRILOK         ;TYPE "WRITE-LOCKED"
7489  033010  000726                        BR      2$              ;TAKE ERROR EXIT
7490                          ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
7491  033012  112765  000103  000001  8$:   MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
7492  033020  004737  042510                JSR     PC,DRVCAL       ;SET VOLUME VALID
7493  033024  112765  000113  000001        MOVB    #RECAL,P.CMND(R5)       ;SET RECAL COMMAND
7494  033032  004737  042510                JSR     PC,DRVCAL       ;RECALIBRATE THE DRIVE
7495  033036  112765  000121  000001        MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
7496  033044  105737  003124                TSTB    TYPFMT          ;SEE IF RK07
7497  033050  001004                        BNE     25$             ;BR IF YES
7498  033052  012765  000632  000002        MOV     #632,P.CYLN(R5)
7499  033060  000403                        BR      27$
7500  033062  012765  001456  000002  25$:  MOV     #1456,P.CYLN(R5)
7501  033070  112765  000002  000005  27$:  MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
7502  033076  012765  065636  000010        MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
7503  033104  012765  177774  000012        MOV     #-4,P.WC(R5)    ;READ 4 WORDS
7504  033112  142765  000020  000007        BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
7505  033120  153765  003124  000007        BISB    TYPFMT,P.CS1H(R5)
7506  033126  005737  003150                TST     AUTOFG          ;SEE IF FROM AUTO MODE
7507  033132  001027                        BNE     12$             ;EXIT IF SO
7508  033134  004737  042510                JSR     PC,DRVCAL       ;READ 4 WORDS OF BAD SECTOR FILE
7509  033140  032737  100000  005474        BIT     #ANYDER,RECODE  ;SEE IF DATA ERROR
7510  033146  001402                        BEQ     10$             ;BR IF OK
7511  033150  104401  013141                TYPE    ,BAD632         ;TYPE READ ERROR MESSAGE
7512  033154  022737  177777  065644  10$:  CMP     #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
7513  033162  001013                        BNE     12$             ;BR IF NOT ALL 1'S
7514  033164  113737  005500  007467        MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
7515  033172  152737  000060  007467        BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
7516  033200  104401  007461                TYPE    ,BADDRV         ;TYPE "DRIVE X"
7517  033204  104401  007533                TYPE    ,ALNPAK         ;TYPE "LOADED WITH ALIGN PACK"
7518  033210  000626                        BR      2$              ;TAKE ERROR EXIT
7519                          ;ERROR FREE RETURN
7520  033212  062716  000002          12$:  ADD     #2,(SP)         ;FIX UP RETURN PC
7521  033216  000207                        RTS     PC              ;RETURN
7522
7523
7524
7525                          ;;*****************************************************************
7526                          .SBTTL  CHKDRV - CHECK STATUS OF DRIVE
7527                          ;*THIS SUBROUTINE CHECKS THE DESIRED DRIVE TO DETERMINE
7528                          ;*IF THE DRIVE IS ON-LINE,READY,WRITE-PROTECTED. IF NOT, THE
7529                          ;*APPROPRIATE ERROR MESSAGE IS TYPED,AND THE CPU HALTS
7530                          ;*WHILE MANUAL  INTERVENTION TAKES PLACE TO CORRECT THE
7531                          ;*PROBLEM. THE OPERATOR THEN DEPRESSES 'CONT' TO  PROCEED,
7532                          ;*AND A RETURN IS MADE TO THE ADDRESS LISTED IMMEDIATELY
7533                          ;*AFTER THE CALL. IF THERE ARE NO  PROBLEMS WITH THIS
7534                          ;*DRIVE, A NORMAL  RETURN IS MADE.
7535                          ;*R5 MUST CONTAIN THE ADDRESS OF THE PARAMETER BLOCK,
7536                          ;*AND R2 MUST CONTAIN THE RK06 REGISTER BASE ADDRESS.
7537                          ;*
```

```
                                        ;* CALL:
                                        ;*      JSR     PC,CHKDRV
                                        ;*      ERROR ADDRESS
                                        ;;*****************************************************************
                                        
033220  113765  005500  000000  CHKDRV: MOVB    DRIVE,P.DRVN(R5) ;SET DESIRED DRIVE NUMBER
033226  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
033234  012737  030320  003046          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
033242  012737  000377  005530          MOV     #377,NEWON      ;INITIALIZE ON-LINE INDICATOR
033250  004737  042510                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
                                    ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
033254  012737  044156  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNORMAL DRIVER RETURN ADDRESS
033262  022737  000377  005530          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
033270  001411                          BEQ     1$              ;BR IF NED NOT SET
033272  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
033276  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
033302  110137  011713                  MOVB    R1,DRVNED       ;PUT DRIVE NO. INTO MSG BUFFER
033306  104401  011703                  TYPE    ,NONEXD         ;TYPE NON-EXISTENT DRIVE MESSAGE
033312  000414                          BR      3$              ;SERVICE THE ERROR
                                    ;SEE IF DESIRED DRIVE IS READY
033314  032765  000200  000040  1$:     BIT     #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
033322  001024                          BNE     4$              ;BR IF DRIVE READY
033324  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
033330  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
033334  110137  011561                  MOVB    R1,DRNRDY       ;PUT DRIVE NUMBER IN MESSAGE BUFFER
033340  104401  011551                  TYPE    ,NOTRDY         ;TYPE DRIVE NOT READY MESSAGE
                                    ;SERVICE ALL ERRORS HERE
033344  042762  000100  000000  3$:     BIC     #IE,RKCS1(R2)   ;CLEAR RK06 INTERRUPT ENABLE BIT
033352  000000                          HALT                    ;HALT IS NECESSARY DURING MANUAL
                                                                ;  INTERVENTIONS, TO CHANGE PACK,
                                                                ;  WRITE LOCK,START DRIVE,ETC.
                                                                ;PRESS 'CONT' TO PROCEED !
033354  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
033362  004737  042510                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
033366  017616  000000                  MOV     @(SP),(SP)      ;SET UP ERROR RETURN ADDRESS
033372  000207                          RTS     PC              ;ERROR RETURN
                                    ;SEE IF DRIVE IS WRITE-LOCKED
033374  032765  004000  000040  4$:     BIT     #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
033402  001011                          BNE     5$              ;BR IF WRITE-PROTECTED
033404  113701  005500                  MOVB    DRIVE,R1        ;GET DRIVE NUMBER
033410  152701  000060                  BISB    #'0,R1          ;CONVERT TO ASCII
033414  110137  012010                  MOVB    R1,NODRLK       ;PUT DRIVE NUMBER IN MSG BUFFER
033420  104401  012000                  TYPE    ,NOTLOK         ;TYPE 'DRIVE NOT WRITE-LOCKED''
033424  000747                          BR      3$              ;SERVICE THE ERROR
                                    ;RECALIBRATE DESIRED DRIVE, SET VOLUME VALID
033426  112765  000113  000001  5$:     MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
033434  004737  042510                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
033440  112765  000103  000001          MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
033446  004737  042510                  JSR     PC,DRVCAL       ;DO PACK ACK. (SETS VOLUME VALID)
                                    ;RETURN HERE
033452  062716  000002          7$:     ADD     #2,(SP)         ;FIX UP RETURN ADDRESS ON STACK
033456  000207                          RTS     PC              ;ERROR-FREE RETURN
                                    
                                    
                                    
                                    ;;*****************************************************************
```

```
                                        ;* ALNSEK - SEEK IN SINGLE INCREMENTS FROM CYL 0 TO ALIGNMENT CYL (365 OCT).
                                        ;;*****************************************************************
033460  112765  000117  000001  ALNSEK: MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
033466  005065  000002                  CLR     P.CYLN(R5)       ;INIT CYL TO 0
033472  004737  042510          2$:     JSR     PC,DRVCAL        ;SEEK TO THIS CYL
033476  005265  000002                  INC     P.CYLN(R5)       ;INCR CYL NO.
033502  105737  003124                  TSTB    TYPFMT           ;CURRENT DRIVE IS AN RK07 ?
033506  001005                          BNE     3$               ;BRANCH IF IT IS
033510  022765  000366  000002          CMP     #ALNCYL+1,P.CYLN(R5) ;SEE IF 365 REACHED YET
033516  001365                          BNE     2$               ;BR IF NOT YET
033520  000207                          RTS     PC               ;RETURN
033522  022765  000761  000002  3$:     CMP     #761,P.CYLN(R5) ;REACH THE CYLINDER 760 FOR RK07
033530  001360                          BNE     2$               ;CONTINUE
033532  000207                          RTS     PC               ;EXIT


                                        ;;*****************************************************************
                                        ;* WAIT4R - THIS SUBROUTINE UNLOADS HEADS ON THE CURRENT DRIVE, TYPES
                                        ;* "TYPE <R> WHEN READY", AND THEN LOADS THE HEADS WHEN <R> IS TYPED.
                                        ;;*****************************************************************
033534  112765  000107  000001  WAIT4R: MOVB    #UNLOAD,P.CMND(R5) ;SET UNLOAD COMMAND
033542  004737  042510                  JSR     PC,DRVCAL        ;UNLOAD HEADS ON THIS DRIVE
033546  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
033554  004737  042510          1$:     JSR     PC,DRVCAL        ;READ STATUS OF DRIVE
033560  032765  000040  000044          BIT     #S.HDHM,P.A01(R5) ;SEE IF HEADS UNLOADED YET
033566  001772                          BEQ     1$               ;BR IF NOT YET
033570  104401  012610          2$:     TYPE    ,RWNRDY          ;TYPE "TYPE <R> WHEN READY"
033574  004737  030536                  JSR     PC,PREPKB        ;PREPARE FOR KBD INPUT
033600  005737  005522          4$:     TST     INTCHR           ;SEE IF ANY INPUT YET
033604  001775                          BEQ     4$               ;BR IF NOT YET
033606  022737  000122  005522          CMP     #'R,INTCHR       ;SEE IF <R> TYPED
033614  001403                          BEQ     6$               ;BR IF <R> TYPED
033616  004737  030556                  JSR     PC,ECOBAD        ;ECHO BAD INPUT
033622  000762                          BR      2$               ;GO ASK AGAIN
033624  112765  000111  000001  6$:     MOVB    #SRTSPL,P.CMND(R5) ;SET START SPINDLE COMMAND
033632  004737  042510                  JSR     PC,DRVCAL        ;START SPINDLE AND LOAD HEADS
033636  000207                          RTS     PC               ;RETURN


                                        ;;*****************************************************************
                                        ;*SETTUP - SET UP FOR LOOP ON ERROR
                                        ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
                                        ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
                                        ;;*****************************************************************
033640  011637  001076          SETTUP: MOV     (SP),@#STACK-2   ;MOVE RETURN PC ON STACK
033644  012706  001076                  MOV     #STACK-2,SP      ;RE-INIT THE STACK POINTER
033650  005037  005474                  CLR     RECODE           ;CLEAR ERROR RECOVERY FLAGS
033654  105037  003126                  CLRB    ERRCNT           ;CLEAR RETRY ERROR COUNT
033660  012705  002630                  MOV     #PARM0,R5        ;SET PARAM BLK ADRS
033664  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
033672  004737  042510                  JSR     PC,DRVCAL        ;CLEAR THE SUBSYSTEM
033676  012737  000000  177776          MOV     #PR0,@#PS        ;RE-ESTABLISH PRIORITY 0
033704  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.

033706  010146                  CHKITR: MOV     R1,-(SP)        ;SAVE R1
033710  112737  000144  001115          MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
033716  105737  003135                  TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
033722  001006                          BNE     3$              ;BR IF YES
033724  105737  003116                  TSTB    MDFLAG          ;SEE IF 200 START
033730  001007                          BNE     4$              ;BR IF NOT
033732  005737  001326                  TST     $PASS           ;SEE IF FIRST PASS
033736  001004                          BNE     4$              ;BR IF NOT
033740  012737  000001  001304  3$:     MOV     #1,$TIMES       ;SET UP FOR 1 ITERATION
033746  000410                          BR      1$              ;GO RUN DUAL-ACCESS TEST
033750  113701  001102          4$:     MOVB    $TSTNM,R1       ;GET CURRENT TEST NUMBER
033754  005301                          DEC     R1              ;DECREMENT BY 1
033756  006301                          ASL     R1              ;DOUBLE IT, TO GET TEST LIST INDEX
033760  016137  005630  001304          MOV     TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
033766  001403                          BEQ     2$              ;BR IF TEST SHOULD BE SKIPPED
033770  062766  000004  000002  1$:     ADD     #4,2(SP)        ;ADJUST RETURN PC TO RUN THIS TEST
033776  012601                  2$:     MOV     (SP)+,R1        ;RESTORE R1
034000  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 BUFF0, TERMINATED BY A NULL (0) BYTE.
                                        ;*RUB-OUT AND (^U) FEATURES ARE PROVIDED.
                                        ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
                                        ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
                                        ;*TAKEN IF (^C) IS TYPED, THE SECOND IS TAKEN IF (^Z) IS
                                        ;*TYPED, AND THE THIRD IS TAKEN IF (^U) OR INVALID INPUT IS TYPED.
                                        ;*IF (^G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
                                        ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (^G) RETURN
                                        ;*IS TAKEN.
                                        ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
                                        ;*IN R0.
                                        ;*
                                        ;*  CALL -      JSR     PC,RDCHRS
                                        ;*              <CONTROL-C RETURN ADDRESS>
                                        ;*              <CONTROL-Z RETURN ADDRESS>
                                        ;*              <CONTROL-U OR ERROR RETURN ADDRESS>
                                        ;*              RETURN
                                        ;;*****************************************************************
```

```
7706  034002                        RDCHRS:
7707  034002  010146                        MOV     R1,-(SP)        ;SAVE R1
7708  034004  010246                        MOV     R2,-(SP)        ;SAVE R2
7709  034006  005000                        CLR     R0              ;INITIALIZE CHARACTER COUNT
7710  034010  005001                        CLR     R1              ;INITIALIZE RUB-OUT INDICATOR
7711                                ;READ A CHARACTER
7712  034012  104406                2$:     RDCHR                   ;READ A CHARACTER
7713  034014  112602                        MOVB    (SP)+,R2        ;GET CHARACTER INTO R2
7714                                ;CHECK FOR (^C)
7715  034016  122702  000003               CMPB    #003,R2         ;SEE IF (^C) TYPED
7716  034022  001006                        BNE     4$              ;BR IF NOT (^C)
7717  034024  104401  013205                TYPE    ,CNTRLC         ;ECHO (^C)
7718  034030  017666  000004  000004 3$:    MOV     @4(SP),4(SP)    ;PUT RETURN ADDRESS ON STACK
7719  034036  000523                        BR      24$             ;BR TO TAKE EXIT
7720                                ;CHECK FOR (^Z)
7721  034040  122702  000032        4$:     CMPB    #032,R2         ;SEE IF (^Z) TYPED
7722  034044  001006                        BNE     6$              ;BR IF NOT (^Z)
7723  034046  104401  013212                TYPE    ,CNTRLZ         ;ECHO (^Z)
7724  034052  062766  000002  000004        ADD     #2,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADR.
7725  034060  000763                        BR      3$              ;BR TO TAKE (^Z) EXIT
7726                                ;CHECK FOR (^U)
7727  034062  122702  000025        6$:     CMPB    #025,R2         ;SEE IF (^U) TYPED
7728  034066  001006                        BNE     8$              ;BR IF NOT (^U)
7729  034070  104401  013224                TYPE    ,CNTRLU         ;ECHO (^U)
7730  034074  062766  000004  000004 7$:    ADD     #4,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
7731  034102  000752                        BR      3$              ;BR TO TAKE (^U) EXIT
7732                                ;CHECK FOR (^G)
7733  034104  122702  000007        8$:     CMPB    #007,R2         ;SEE IF (^G) TYPED
7734  034110  001005                        BNE     9$              ;BR IF NOT (^G)
7735  034112  104401  013231                TYPE    ,CNTRLG         ;ECHO (^G)
7736  034116  004737  031066                JSR     PC,GTSWRG       ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
7737  034122  000764                        BR      7$              ;TAKE (^U) RETURN
7738                                ;CHECK FOR RUB-OUT (DELETE)
7739  034124  122702  000177        9$:     CMP3    #177,R2         ;SEE IF RUB-OUT (DEL) TYPED
7740  034130  001020                        BNE     14$             ;BR IF NOT RUB-OUT
7741  034132  005700                        TST     R0              ;CHECK THE CHARACTER COUNT
7742  034134  001726                        BEQ     2$              ;BR IF COUNT = 0
7743  034136  005701                        TST     R1              ;CHECK THE RUB-OUT INDICATOR
7744  034140  001003                        BNE     11$             ;BR IF WE HAD A PREVIOUS RUB-OUT
7745  034142  005201                        INC     R1              ;SET RUB-OUT INDICATOR
7746  034144  104401  013242                TYPE    ,BKSLSH         ;TYPE A BACK-SLASH (\)
7747  034150  005037  005534        11$:    CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
7748  034154  005300                        DEC     R0              ;DECREMENT COUNT
7749  034156  116037  005264  005534        MOVB    BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
7750  034164  104401  005534                TYPE    ,SCRACH         ;ECHO CHARACTER TO BE DELETED
7751  034170  000710                        BR      2$              ;GO READ ANOTHER CHARACTER
7752  034172  005701                14$:    TST     R1              ;CHECK THE RUB-OUT INDICATOR
7753  034174  001403                        BEQ     16$             ;BR IF INDICATOR IS NOT SET
7754  034176  104401  013242                TYPE    ,BKSLSH         ;TYPE A BACK-SLASH
7755  034202  005001                        CLR     R1              ;CLEAR THE RUB-OUT INDICATOR
7756                                ;CHECK FOR CARRIAGE RETURN
7757  034204  122702  000015        16$:    CMPB    #015,R2         ;SEE IF <CR> TYPED
7758  034210  001426                        BEQ     19$             ;BR IF <CR>
7759                                ;HANDLE POSSIBLE DIGIT
7760  034212  005037  005534                CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
7761  034216  110237  005534                MOVB    R2,SCRACH       ;GET THIS CHARACTER INTO BUFFER
```

```
 7762  034222  104401  005534            TYPE    ,SCRACH          ;ECHO THE CHARACTER TYPED
 7763  034226  110260  005264            MOVB    R2,BUFFO(R0)     ;PUT CHARACTER INTO BUFFER
 7764  034232  005200                    INC     R0               ;INCREMENT CHARACTER COUNTER
 7765  034234  022700  000120            CMP     #80.,R0          ;SEE IF TOO MANY CHARACTERS TYPED
 7766  034240  001264                    BNE     2$               ;BR IF NOT TOO MANY
 7767  034242  104401  001315            TYPE    ,$CRLF           ;TYPE <CR> AND <LF>
 7768  034246  112760  000000  005264    MOVB    #0,BUFFO(R0)  ;PUT TERMINATING NULL INTO BUFFER
 7769  034254  104401  005264            TYPE    ,BUFFO           ;ECHO INPUT STRING
 7770  034260  104401  001314            TYPE    ,$QUES           ;TYPE <?>,<CR>,<LF>
 7771  034264  000703                    BR      7$               ;TAKE ERROR EXIT FROM RDCHRS
 7772  034266  104401  001315    19$:    TYPE    ,$CRLF           ;TYPE <CR>,<LF>
 7773  034272  112760  000000  005264    MOVB    #0,BUFFO(R0)     ;PUT TERMINATING NULL INTO BUFFER
 7774  034300  062766  000006  000004    ADD     #6,4(SP)         ;FIX UP RETURN ADDRESS
 7775  034306  012602            24$:    MOV     (SP)+,R2         ;RESTORE R2
 7776  034310  012601                    MOV     (SP)+,R1         ;RESTORE R1
 7777  034312  000207                    RTS     PC               ;SUBROUTINE EXIT
 7778
 7779
 7780
 7781                                    ;;*****************************************************************
 7782                                    .SBTTL  TYPTST - TYPE CURRENT TEST AND ITERATION NUMBER
 7783                                    ;*THIS SUBROUTINE TYPES : "XX     YYYYYY", WITH NO <CR>
 7784                                    ;*OR <LF>. XX IS THE NUMBER OF THE CURRENT TEST AND
 7785                                    ;*YYYYYY IS THE NUMBER OF ITERATIONS FOR THAT TEST.
 7786                                    ;*R1 MUST CONTAIN THE INDEX INTO THE TEST LIST FOR
 7787                                    ;*THE CURRENT TEST.
 7788                                    ;* CALL -   JSR    PC,TYPTST
 7789                                    ;;*****************************************************************
 7790
 7791  034314  104401  013253    TYPTST: TYPE    ,SPACE1          ;TYPE A SPACE
 7792  034320  010146                    MOV     R1,-(SP)         ;PUT INDEX ONTO STACK
 7793  034322  006216                    ASR     (SP)             ;DIVIDE BY 2
 7794  034324  005216                    INC     (SP)             ;INCREMENT TO GET TEST NO.
 7795  034326  104403                    TYPOS                    ;TYPE TEST NO. IN OCTAL
 7796  034330     002                    .BYTE   2                ;TYPE 2 DIGITS
 7797  034331     000                    .BYTE   0                ;SUPPRESS LEADING ZEROS
 7798  034332  104401  013246            TYPE    ,SPACE6          ;TYPE 6 SPACES
 7799  034336  016146  005630            MOV     TSTLST(R1),-(SP) ;PUT CURRENT ITERATION NO. ONTO STACK
 7800  034342  104403                    TYPOS                    ;TYPE ITERATION NO. IN OCTAL
 7801  034344     006                    .BYTE   6                ;TYPE 6 DIGITS
 7802  034345     000                    .BYTE   0                ;SUPPRESS LEADING ZEROS
 7803  034346  000207                    RTS     PC               ;RETURN
 7804
 7805
 7806                                    ;;*****************************************************************
 7807                                    .SBTTL  TYPPRM - TYPE CURRENT PARAMETER
 7808                                    ;*THIS SUBROUTINE TYPES THE LINE : XX=YYYYYYYY ,
 7809                                    ;*WHERE XX IS THE CURRENT PARAMETER MNEMONIC, AND
 7810                                    ;*YYYYYYYY IS ITS VALUE IN OCTAL DIGITS, WITH LEADING
 7811                                    ;*ZEROS SUPPRESSED. NOTE: <CR> AND <LF> ARE NOT DONE.
 7812                                    ;*ON ENTRY, R1 MUST CONTAIN THE INDEX INTO THE
 7813                                    ;*PARAMETER TABLES, FOR THE CURRENT PARAMETER.
 7814                                    ;* CALL -      JSR     PC,TYPPRM
 7815                                    ;;*****************************************************************
 7816
 7817  034350  016137  006010  013256    TYPPRM: MOV     PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
```

```
 7818  034356  104401  013256                  TYPE    ,PRMBUF          ;TYPE "XX="
 7819  034362  016137  005660  005470          MOV     PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
 7820  034370  005037  005472                  CLR     HIGOCT           ;CLEAR HIGH BINARY BITS
 7821  034374  022761  040515  006010          CMP     #'MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
 7822  034402  001003                          BNE     1$               ;BR IF NOT (MA)
 7823  034404  016137  005662  005472          MOV     PRMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
 7824  034412  012746  005470          1$:     MOV     #LOWOCT,-(SP)    ;PUT ADDRESS OF BINARY VALUE ON STACK
 7825  034416  004737  055414                  JSR     PC,@#$DB2O       ;CONVERT BINARY TO OCTAL ASCII
 7826  034422  004737  055730                  JSR     PC,@#$SUPRS      ;TYPE YYYYYYYY, SUPPRESSING LEADING 0'S
 7827  034426  000207                          RTS     PC               ;RETURN
 7828
 7829
 7830
 7831                                  ;;*****************************************************************
 7832                                  .SBTTL  TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15
 7833                                  ;*THIS SUBROUTINE TYPES THE LINE : WORD XX = YYYYYY,
 7834                                  ;*WHERE XX IS THE NUMBER OF THE CURRENT WORD OF
 7835                                  ;*USER-DEFINED PATTERN 15, AND YYYYYY IS ITS 16-BIT
 7836                                  ;*VALUE, IN OCTAL DIGITS. <CR> AND <LF> ARE NOT TYPED.
 7837                                  ;*ON ENTRY, R1 MUST CONTAIN THE INDEX OF THE CURRENT
 7838                                  ;*WORD IN THE PATTERN 15 TABLE.
 7839                                  ;*  CALL -     JSR       PC,TYPPAT
 7840                                  ;;*****************************************************************
 7841
 7842  034430                          TYPPAT:
 7843  034430  104401  013262                  TYPE    ,WORD SP         ;TYPE "WORD "
 7844  034434  010146                          MOV     R1,-(SP)         ;PUT INDEX ONTO STACK
 7845  034436  006216                          ASR     (SP)             ;DIVIDE BY TWO FOR WORD NO.
 7846  034440  104403                          TYPOS                    ;GO TYPE WORD NUMBER
 7847  034442     002                          .BYTE   2                ;DIGIT COUNT = 2 FOR TYPOC
 7848  034443     001                          .BYTE   1                ;TELL TYPOS TO TYPE LEADING ZEROS
 7849  034444  104401  013270                  TYPE    ,EQUALS          ;TYPE " = "
 7850  034450  016146  006776                  MOV     PAT15(R1),-(SP) ;GET BINARY VALUE OF THIS WORD
 7851  034454  104403                          TYPOS                    ;TYPE VALUE IN OCTAL
 7852  034456     006                          .BYTE   6                ;TYPE 6 DIGITS
 7853  034457     001                          .BYTE   1                ;TYPE LEADING ZEROS
 7854  034460  000207                          RTS     PC               ;RETURN
 7855
 7856
 7857
 7858                                  ;;*****************************************************************
 7859                                  .SBTTL  MODP15 - MODIFY USER-DEFINED PATTERN 15
 7860                                  ;*THIS SUBROUTINE DETERMINES WHETHER OR NOT USER-DEFINED
 7861                                  ;*DATA PATTERN 15 SHOULD BE OPENED FOR MODIFICATION, AND
 7862                                  ;*IF SO, IT REQUESTS THE NEW VALUES AND CONTROLS THEIR
 7863                                  ;*INPUT ON THE TTY. IT CHECKS VALUES FOR VALIDITY, AND
 7864                                  ;*LOADS THE 16 WORDS INTO THE PATTERN 15 TABLE (PAT15).
 7865                                  ;*IF THE OPERATOR ANSWERS THE REQUEST FOR A PATTERN WORD
 7866                                  ;*WITH EXCLAMATION POINT (!) (CR), THE SUBROUTINE
 7867                                  ;*PROPAGATES THE LAST WORD TYPED INTO ALL REMAINING
 7868                                  ;*WORDS OF THE PATTERN 15 TABLE.
 7869                                  ;*  CALL -    JSR      PC,MODP15
 7870                                  ;;*****************************************************************
 7871
 7872  034462                          MODP15:
 7873  034462  010046                          MOV     R0,-(SP)         ;SAVE R0
```

```
                                        MOV     R1,-(SP)        ;SAVE R1
                                        MOV     R2,-(SP)        ;SAVE R2
                                ;SEE IF PARAMETER IS PT
                                        CMP     #''PT,PRMNEM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
                                        BNE     22$             ;BR IF NOT (PT)
                                ;SEE IF PATTERN 15 IS SPECIFIED
                                        BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
                                        BEQ     22$             ;BR IF NOT SPECIFIED
                                ;SEE IF PATTERN 15 SHOULD BE MODIFIED
                                4$:     TYPE    ,MDFY15         ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
                                        JSR     PC,RDCHRS       ;READ RESPONSE
                                        24$                     ;(^C) RETURN ADDRESS
                                        26$                     ;(^Z) RETURN ADDRESS
                                        8$                      ;(^U) OR ERROR RETURN ADDRESS
                                        TST     R0              ;SEE IF NULL INPUT
                                        BEQ     22$             ;BR IF MODIFICATION NOT REQUESTED
                                        CMP     #'M,BUFF0       ;SEE IF (M) TYPED
                                        BEQ     6$              ;BR IF MODIFICATION REQUESTED
                                        TYPE    ,BUFF0          ;ECHO BAD INPUT
                                        TYPE    ,$QUES
                                        BR      4$              ;GO ASK AGAIN
                                ;MODIFY PATTERN 15
                                6$:     TYPE    ,SELP15         ;TYPE 'MODIFY PATTERN 15''
                                        CLR     R1              ;INITIALIZE WORD INDEX
                                8$:     JSR     PC,TYPPAT       ;TYPE CURRENT WORD AND VALUE
                                        TYPE    ,SPACE1         ;TYPE A SPACE
                                        TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
                                ;READ AND CHECK INPUT, IF ANY
                                        JSR     PC,RDCHRS       ;READ NEW DATA PATTERN WORD
                                        24$                     ;(^C) RETURN ADDRESS FOR RDCHRS
                                        26$                     ;(^Z) RETURN ADDRESS FOR RDCHRS
                                        8$                      ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
                                        TST     R0              ;SEE IF ANY INPUT
                                        BNE     12$             ;BR IF ANY INPUT
                                10$:    ADD     #2,R1           ;INCREMENT WORD INDEX
                                        CMP     #32.,R1         ;SEE IF ALL DONE
                                        BEQ     22$             ;BR IF DONE, TO RETURN
                                        BR      8$              ;CONTINUE WITH NEXT WORD
                                12$:    CMP     #'!,BUFF0       ;SEE IF (!) TYPED
                                        BEQ     16$             ;BR TO PROPAGATE CURRENT VALUE
                                        CLR     R2              ;INIT. (!) INDICATOR
                                        CMPB    #'!,BUFF0-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
                                        BNE     14$             ;BR IF NOT (!)
                                        CLRB    BUFF0-1(R0)     ;INSERT TERMINATOR BYTE
                                        DEC     R0              ;DECREMENT CHAR COUNT
                                        INC     R2              ;SET (!) INDICATOR
                                14$:    CMP     #6,R0           ;SEE HOW MANY CHARS NOW
                                        BLT     20$             ;BR IF TOO MANY
                                        MOV     #BUFF0,-(SP)    ;GET BUF. ADDR. ON STACK FOR OCTBIN
                                        JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
                                        20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
                                        MOV     (SP)+,R0        ;GET BINARY VALUE
                                        TST     $HIOCT          ;SEE IF VALUE EXCEEDS 16 BITS
                                        BNE     20$             ;BR TO ECHO BAD INPUT
                                        MOV     R0,PAT15(R1)    ;PUT NEW WORD VALUE INTO TABLE
                                        TST     R2              ;SEE IF (!) WAS TYPED
```

```
7930  034720  001735                        BEQ     10$             ;BR IF (!) WAS NOT TYPED
7931                                ;PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE
7932  034722  022701  000036        16$:    CMP     #30.,R1         ;SEE IF ALL DONE YET
7933  034726  001413                        BEQ     22$             ;BR IF DONE
7934  034730  016161  006776  007000        MOV     PAT15(R1),PAT15+2(R1) ;PROPAGATE TO NEXT WORD
7935  034736  062701  000002                ADD     #2,R1           ;INCREMENT WORD INDEX
7936  034742  000767                        BR      16$             ;LOOP UNTIL DONE
7937                                ;ECHO BAD INPUT
7938  034744  104401  005264        20$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
7939  034750  104401  001314                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
7940  034754  000702                        BR      8$              ;BR TO ASK AGAIN
7941                                ;NORMAL RETURN
7942  034756  012602        22$:    MOV     (SP)+,R2        ;RESTORE R2
7943  034760  012601                MOV     (SP)+,R1        ;RESTORE R1
7944  034762  012600                MOV     (SP)+,R0        ;RESTORE R0
7945  034764  000207                RTS     PC              ;RETURN
7946                                ;(^C) RETURN
7947  034766  012766  014374  000006 24$:    MOV     #DRVTST,6(SP)   ;PC FOR (^C) RETURN
7948  034774  000770                BR      22$             ;BR TO RETURN
7949                                ;(^Z) RETURN
7950  034776  012766  015760  000006 26$:    MOV     #ASKMDE,6(SP)   ;PC FOR (^Z) RETURN
7951  035004  000764                BR      22$             ;BR TO RETURN
7952
7953
7954
7955                                ;;*****************************************************************
7956                                .SBTTL  CHKPRM - CHECK VALUE OF INPUT PARAMETER
7957                                ;*THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
7958                                ;*HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
7959                                ;*LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
7960                                ;*INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
7961                                ;*TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
7962                                ;*IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
7963                                ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
7964                                ;*THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
7965                                ;*VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
7966                                ;* CALL -      JSR     PC,CHKPRM
7967                                ;*             <ERROR RETURN ADDRESS>
7968                                ;*             RETURN
7969                                ;;*****************************************************************
7970
7971  035006  104407        CHKPRM: SAVREG                  ;SAVE R0-R5
7972  035010  010102                MOV     R1,R2           ;GET COPY OF INDEX
7973  035012  006302                ASL     R2              ;DOUBLE IT
7974  035014  022761  040515  006010        CMP     #'MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
7975  035022  001422                BEQ     20$             ;BR IF (MA)
7976  035024  005737  005472                TST     HIGOCT          ;SEE IF HIGH BITS = 0
7977  035030  001403                BEQ     18$             ;BR IF ZERO
7978  035032  017616  000000        16$:    MOV     @(SP),(SP)      ;GET ERROR RETURN PC
7979  035036  000475                BR      30$             ;GO TO RETURN
7980                                ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
7981  035040  023762  005470  005734 18$:    CMP     LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
7982  035046  103771                BLO     16$             ;BR IF INPUT VALUE TOO SMALL
7983  035050  023762  005470  005736        CMP     LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
7984  035056  101365                BHI     16$             ;BR IF INPUT VALUE IS TOO LARGE
7985                                ;UPDATE 16-BIT PARAMETER VALUE IN LIST
```

```
 7986  035060  013761  005470  005660          MOV     LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
 7987  035066  000457                          BR      28$               ;BR TO RETURN
 7988                                  ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
 7989  035070  032737  000001  005470  20$:     BIT     #BIT0,LOWOCT      ;SEE IF MA IS ODD
 7990  035076  001355                          BNE     16$               ;BR IF MA IS ODD
 7991  035100  023762  005472  005736          CMP     HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
 7992  035106  103751                          BLO     16$               ;BR IF HIGH WORD IS TOO SMALL
 7993  035110  001004                          BNE     24$               ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
 7994  035112  023762  005470  005734          CMP     LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
 7995  035120  103744                          BLO     16$               ;BR IF LOW WORD IS TOO SMALL
 7996  035122  023762  005472  005742  24$:     CMP     HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
 7997  035130  101340                          BHI     16$               ;BR IF HIGH WORD TOO BIG
 7998  035132  001004                          BNE     26$               ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
 7999  035134  023762  005470  005740          CMP     LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
 8000  035142  101333                          BHI     16$               ;BR IF LOW WORD IS TOO LARGE
 8001                                  ;UPDATE 32-BIT PARAMETER VALUE IN LIST
 8002  035144  013761  005470  005660  26$:     MOV     LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
 8003  035152  013761  005472  005662          MOV     HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
 8004                                  .COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
 8005  035160  004737  035236                  JSR     PC,MXWRDC         ;GET WORD COUNT IN R1-R0
 8006  035164  104401  010405                  TYPE    ,MAWRDC           ;TYPE 'MAX WORD COUNT = ''
 8007  035170  010037  003172                  MOV     R0,SUMLO1
 8008  035174  010137  003174                  MOV     R1,SUMHI1
 8009  035200  012746  003172                  MOV     #SUMLO1,-(SP)     ;PUT POINTER ON STACK
 8010  035204  004737  055414                  JSR     PC,$DB2O          ;CONVERT TO OCTAL
 8011  035210  004737  055730                  JSR     PC,$SUPRS         ;TYPE IT
 8012  035214  104401  001315                  TYPE    ,$CRLF            ;TYPE <CR>,<LF>
 8013  035220  062766  000002  000014          ADD     #2,14(SP)         ;INCREMENT R1 INDEX
 8014  035226  062716  000002          28$:     ADD     #2,(SP) ;GET NORMAL RETURN PC
 8015  035232  104410                  30$:     RESREG                    ;RESTORE R0-R5
 8016  035234  000207                          RTS     PC                ;RETURN
 8017
 8018
 8019
 8020                                  ;;*****************************************************************
 8021                                  ;*MXWRDC - GET MAX WORD COUNT FOR CURRENT MEM LIMITS AND CURRENT MA,
 8022                                  ;*AND LEAVE THE WORD COUNT IN R1-R0. NO REGISTERS ARE SAVED.
 8023                                  ;;*****************************************************************
 8024  035236  013700  005574  MXWRDC: MOV     MAHILM,R0         ;GET LO BITS OF MEM LIM
 8025  035242  013701  005576          MOV     MAHILM+2,R1       ;HI BITS OF MEM LIM
 8026  035246  163700  005600          SUB     MA,R0             ;SUBTRACT MA FROM MAHILM
 8027  035252  005601                  SBC     R1
 8028  035254  163701  005602          SUB     MA+2,R1
 8029  035260  100413                  BMI     27$               ;IF NEGATIVE, RETURN
 8030  035262  000241                  CLC                       ;DIVIDE BY 2 TO GET WORDS
 8031  035264  006001                  ROR     R1
 8032  035266  006000                  ROR     R0
 8033  035270  062700  000001          ADD     #1,R0             ;INCREMENT BY 1 WORD
 8034  035274  005501                  ADC     R1
 8035  035276  005701                  TST     R1
 8036  035300  001403                  BEQ     27$               ;BR IF WORD COUNT < 65,536 DFC
 8037  035302  005000                  CLR     R0                ;MAKE WORD COUNT = 65,536
 8038  035304  012701  000001          MOV     #1,R1
 8039  035310  000207          27$:     RTS     PC                ;RETURN
 8040
 8041
```

```
                                ;;*****************************************************************
                                .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
                                ;*THIS SUBROUTINE TYPES "DRIVE SER. NO.  XXX" (IN DECIMAL), WITH LEADING
                                ;*ZEROS SUPPRESSED.
                                ;;*****************************************************************
035312  104407                  DRVSER: SAVREG                  ;SAVE R0-R5
035314  004737  032250                  JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
035320  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
035326  004737  042510                  JSR     PC,DRVCAL       ;READ STATUS OF THIS DRIVE
035332  104401  011074                  TYPE    ,DRIV           ;TYPE "DRIVE"
035336  104401  011110                  TYPE    ,SERNM          ;TYPE " SER. NO.  "
035342  016501  000054                  MOV     P.A11(R5),R1    ;GET "A" STATUS BYTE 11
035346  012704  055516                  MOV     #$OCTVL,R4      ;GET ADDR OF CHAR BUFFER
035352  010446                          MOV     R4,-(SP)        ;STORE IT ON STACK FOR $SUPRS
035354  012703  000003                  MOV     #3,R3           ;INIT CHAR COUNT
035360  006101                          ROL     R1              ;INITIALIZE BIT POSITIONS
035362  006101                          ROL     R1
035364  006101                  4$:     ROL     R1              ;GET NEXT 4 BITS
035366  006101                          ROL     R1
035370  006101                          ROL     R1
035372  006101                          ROL     R1
035374  010100                          MOV     R1,R0           ;GET A WORKING COPY
035376  042700  177760                  BIC     #177760,R0      ;CLEAR ALL BUT LOW 4 BITS
035402  052700  000060                  BIS     #'0,R0          ;CONVERT A DIGIT TO ASCII
035406  110024                          MOVB    R0,(R4)+        ;PUT ASCII DIGIT INTO CHAR BUFFER
035410  005303                          DEC     R3              ;DECREMENT CHAR COUNT
035412  001364                          BNE     4$              ;BR IF NOT 3 CHARS YET
035414  105014                          CLRB    (R4)            ;INSERT NULL TERMINATOR
035416  004737  055730                  JSR     PC,@#$SUPRS     ;TYPE DRIVE SER. NUMBER
035422  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
035426  104410                          RESREG                  ;RESTORE R0-R5
035430  000207                          RTS     PC              ;RETURN
```

```
                                        ;;*****************************************************************
                                        .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
                                        ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXXX" (IN OCTAL),
                                        ;*WITH LEADING ZEROS SUPPRESSED.
                                        ;;*****************************************************************
035432  004737  032250          CRTSER: JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
035436  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
035444  105737  003125                  TSTB    FORMAT          ;CHECK THE ACTUAL FORMAT
035450  001402                          BEQ     4$              ;BR IF 22 SECTORS
035452  105265  000004                  INCB    P.SECT(R5)      ;IF 20 SECTORS, READ SECTOR 1
035456  112765  000121  000001  4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
035464  013765  020000  000002          MOV     LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)/1456
035472  112765  000002  000005          MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
035500  012765  C65636  000010          MOV     #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
035506  012765  177776  000012          MOV     #-2,P.WC(R5)    ;SET WORD COUNT TO READ 2 WORDS
035514  004737  042510                  JSR     PC,DRVCAL       ;READ SERIAL NO. IN BSF
035520  104401  011101                  TYPE    ,CART           ;TYPE   "CART."
035524  104401  011110                  TYPE    ,SERNM          ;TYPE "SER. NO.  "
035530  012746  065636                  MOV     #RWBUF,-(SP)    ;GET POINTER FOR $DB2O
035534  004737  055414                  JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL
035540  004737  055730                  JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
035544  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
035550  000207                          RTS     PC              ;RETURN


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

035552  010046                  STALL:  MOV     R0,-(SP)        ;SAVE R0
035554  010146                          MOV     R1,-(SP)        ;SAVE R1
035556  032777  000400  143354          BIT     #BIT08,@SWR     ;APPLY RANDOM STALL ?
035564  001407                          BEQ     1$              ;BR IF NOT RANDOM
035566  004737  054612                  JSR     PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
035572  013700  054712                  MOV     $LONUM,R0       ;GET IT INTO R0
035576  006200                          ASR     R0              ;SCALE IT DOWN
035600  006200                          ASR     R0
035602  000403                          BR      2$              ;GO STALL WITH RANDOM NO.
035604  013700  005502          1$:     MOV     STALLS,R0       ;GET REQUESTED NO. OF STALLS
035610  001406                          BEQ     6$              ;RETURN IF NO STALL REQUIRED
035612  012701  000016          2$:     MOV     #14.,R1         ;SET CONSTANT FOR 40 US
035616  005301                  4$:     DEC     R1              ;INNER LOOP COUNTER
035620  001376                          BNE     4$              ;INNER LOOP BR
035622  005300                          DEC     R0              ;OUTER LOOP COUNTER
035624  001372                          BNE     2$              ;OUTER LOOP BR
035626  012601                  6$:     MOV     (SP)+,R1        ;RESTORE R1
035630  012600                          MOV     (SP)+,R0        ;RESTORE R0
035632  000207                          RTS     PC              ;RETURN
```

```
 8134                                   ;;*****************************************************************
 8135                                   .SBTTL  RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
 8136                                   ;*THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
 8137                                   ;*ADDRESS, AND LEAVES IT IN "CYLNDR". IT REQUIRES THE SYSMAC
 8138                                   ;*SUBROUTINE $RAND.
 8139                                   ;* CALL -   JSR     PC,RNDADR
 8140                                   ;;*****************************************************************
 8141  035634  004737  054612           RNDADR: JSR     PC,$RAND        ;GENERATE 2 16-BIT RANDOM NUMBERS
 8142  035640  013737  054712  005504           MOV     $LONUM,CYLNDR   ;GET A RANDOM NUMBER
 8143  035646  042737  177000  005504           BIC     #177000,CYLNDR  ;SCALE IT TO 9 BITS
 8144  035654  023737  020000  005504           CMP     LSTCYL,CYLNDR   ;SEE IF CYL IS TOO BIG
 8145  035662  002003                           BGE     2$              ;BR IF CYL IS OK
 8146  035664  043737  020002  005504           BIC     HOLD1,CYLNDR    ;SCALE DOWN TO A VALID CYLINDER
 8147  035672  000207                   2$:     RTS     PC              ;RETURN
 8148
 8149
 8150
 8151                                   ;;*****************************************************************
 8152                                   ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF R0 INTO ALL
 8153                                   ;*256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
 8154                                   ;;*****************************************************************
 8155  035674  104407                   LODSEC: SAVREG                  ;SAVE R0-R5
 8156  035676  012701  065636                   MOV     #RWBUF,R1       ;GET ADDRESS OF DATA BUF INTO R1
 8157  035702  010021                   2$:     MOV     R0,(R1)+        ;PUT WORD INTO BUFFER
 8158  035704  020127  066636                   CMP     R1,#RWBUF+512.  ;SEE IF 256 WORDS WRITTEN YET
 8159  035710  001374                           BNE     2$              ;BR IF NOT DONE YET
 8160  035712  104410                           RESREG                  ;RESTORE R0-R5
 8161  035714  000207                           RTS     PC              ;RETURN
 8162
 8163
 8164
 8165                                   ;;*****************************************************************
 8166                                   ;* TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
 8167                                   ;*FC, FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
 8168                                   ;*TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
 8169                                   ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.
 8170                                   ;;*****************************************************************
 8171  035716  104407                   TRKCHK: SAVREG                  ;SAVE R0-R5
 8172  035720  016546  000004                   MOV     P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
 8173  035724  105065  000005                   CLRB    P.TRCK(R5)      ;CLEAR THE TRACK NO.
 8174                                   ;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
 8175  035730  116500  000005           2$:     MOVB    P.TRCK(R5),R0   ;GET TRACK NO. INTO R0
 8176  035734  062700  000100                   ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
 8177  035740  004737  035674                   JSR     PC,LODSEC       ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
 8178  035744  112765  000131  000001           MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
 8179  035752  004737  042510                   JSR     PC,DRVCAL       ;PERFORM THE WRITE CHECK
 8180  035756  105265  000005                   INCB    P.TRCK(R5)      ;INCREMENT THE TRACK NO.
 8181  035762  122765  000003  000005           CMPB    #3,P.TRCK(R5)   ;SEE IF DONE WITH ALL TRACKS
 8182  035770  001357                           BNE     2$              ;BR IF NOT DONE YET
 8183  035772  012665  000004                   MOV     (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
 8184  035776  104410                           RESREG                  ;RESTORE R0-R5
 8185  036000  000207                           RTS     PC              ;RETURN
 8186
 8187
 8188
 8189                                   ;;*****************************************************************
```

```
                                        ;*LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
                                        ;*INTO THE PATTERN 14 TABLE.
                                        ;;*****************************************************************
036002  104407                  LODP14: SAVREG                          ;SAVE R0-R5
036004  012701  000010                  MOV     #8.,R1                  ;INIT LOOP COUNTER TO 8.
036010  012700  006736                  MOV     #PAT14,R0               ;GET ADDRESS OF PATTERN 14 BUFFER
036014  004737  054612          4$:     JSR     PC,$RAND                ;GENERATE 2 16-BIT RANDOM NUMBERS
036020  013720  054712                  MOV     $LONUM,(R0)+            ;PUT ONE NUMBER INTO PATTERN
036024  013720  054710                  MOV     $HINUM,(R0)+            ;PUT OTHER NO. INTO PATTERN
036030  005301                          DEC     R1                      ;SEE IF 16 WORDS LOADED YET
036032  001370                          BNE     4$                      ;BR IF NOT YET
036034  104410                          RESREG                          ;RESTORE R0-R5
036036  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.
                                        ;;*****************************************************************
036040  104407                  FINADR: SAVREG                          ;SAVE R0-R5
036042  013746  005572                  MOV     LASTWC,-(SP)            ;STORE WORD COUNT
036046  005416                          NEG     (SP)                    ;MAKE IT POSITIVE
036050  005046                  18$:    CLR     -(SP)                   ;MAKE ROOM ON STACK
036052  116616  000003                  MOVB    3(SP),(SP)              ;STORE NO. OF SECTORS TRANSFERRED
036056  005066  000002                  CLR     2(SP)                   ;CLEAR LOCATION ON STACK
036062  116566  000004  000002          MOVB    P.SECT(R5),2(SP)        ;STORE STARTING SECTOR
036070  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL SECTOR ADDRESS
036074  005066  000002                  CLR     2(SP)                   ;CLEAR NO. OF TRACKS TRANSFERRED
036100  012700  000026                  MOV     #22.,R0                 ;SET FOR 22 SECTORS
036104  105737  003125                  TSTB    FORMAT                  ;DETERMINE THE FORMAT
036110  001402                          BEQ     19$                     ;BR IF 22 SECTORS
036112  012700  000024                  MOV     #20.,R0                 ;SET FOR 20 SECTORS
036116  020016                  19$:    CMP     R0,(SP)                 ;CHECK FOR SECTOR OVERFLOW
036120  101004                          BHI     20$                     ;NO, CHECK IF SECTOR CORRECT
036122  160016                          SUB     R0,(SP)                 ;DECREMENT SECTOR COUNT BY 20 OR 22
036124  005266  000002                  INC     2(SP)                   ;INCREMENT TRACKS TRANSFERRED
036130  000772                          BR      19$                     ;CHECK FOR SECTOR OVERFLOW
036132  112637  005571          20$:    MOVB    (SP)+,FINSEC            ;STORE FINAL SECTOR
036136  005046                          CLR     -(SP)                   ;MAKE ROOM FOR TRACKS TRANSFERRED
036140  116516  000005                  MOVB    P.TRCK(R5),(SP)         ;STORE STARTING TRACK
036144  066616  000002                  ADD     2(SP),(SP)              ;DETERMINE FINAL TRACK ADDRESS
036150  005066  000002                  CLR     2(SP)                   ;CLEAR FINAL CYLINDER
036154  122716  000003          21$:    CMPB    #3,(SP)                 ;CHECK FOR TRACK OVERFLOW
036160  101005                          BHI     22$                     ;NO, CHECK FINAL TRACK
036162  162716  000003                  SUB     #3,(SP).                ;DECREMENT TRACK COUNT BY 3
036166  005266  000002                  INC     2(SP)                   ;INCR CYL COUNT
036172  000770                          BR      21$                     ;CHECK FOR TRACK OVERFLOW
036174  112637  005570          22$:    MOVB    (SP)+,FINTRK            ;STORE FINAL TRACK
036200  066516  000002                  ADD     P.CYLN(R5),(SP)         ;CALCULATE FINAL CYLINDER
```

```
 8246  036204  011637  005566                  MOV     (SP),FINCYL     ;STORE FINAL  CYLINDER
 8247  036210  005726                          TST     (SP)+           ;CLEAN OFF STACK
 8248  036212  104410                          RESREG                  ;RESTORE R0-R5
 8249  036214  000207                          RTS     PC              ;RETURN
 8250
 8251
 8252
 8253                                  ;;*****************************************************************
 8254                                  .SBTTL  LODBUF - LOAD THE READ/WRITE DATA BUFFER
 8255                                  ;*THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
 8256                                  ;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
 8257                                  ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
 8258                                  ;*OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
 8259                                  ;*IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
 8260                                  ;*OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
 8261                                  ;*ENTRY, ARE LOADED INTO THE BUFFER.
 8262                                  ;;*****************************************************************
 8263
 8264  036216  104407                  LODBUF: SAVREG                  ;SAVE R0-R5
 8265  036220  013702  005564                  MOV     WDSXFR,R2       ;GET NO. OF WORDS
 8266  036224  005737  005536                  TST     PATRN           ;SEE IF QUICK VERIFY DATA TEST DESIRED
 8267  036230  001007                          BNE     3$              ;BR IF NOT QUICK VERIFY
 8268  036232  012700  006036                  MOV     #PAT00,R0       ;SET DATA PATTERN STARTING ADDRESS
 8269  036236  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
 8270  036242  012701  065636                  MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
 8271  036246  000463                          BR      30$             ;PROCEED
 8272  036250  005000                  3$:     CLR     R0              ;INIT PATTERN NUMBER
 8273  036252  032701  000001          4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
 8274  036256  001003                          BNE     6$              ;BR IF THIS BIT IS SET
 8275  036260  005200                          INC     R0              ;INCREMENT PATTERN NO.
 8276  036262  006201                          ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
 8277  036264  000772                          BR      4$              ;BR TO CHECK NEXT BIT
 8278  036266  006300                  6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
 8279  036270  006300                          ASL     R0
 8280  036272  006300                          ASL     R0
 8281  036274  006300                          ASL     R0
 8282  036276  006300                          ASL     R0
 8283  036300  062700  006036                  ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
 8284  036304  012746  000020                  MOV     #16.,-(SP)      ;SET PATTERN WORD COUNT
 8285  036310  013701  005604                  MOV     PMA,R1          ;SET BUFFER ADDRESS
 8286  036314  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
 8287  036320  100036                          BPL     30$             ;BR IF NOT PRESENT
 8288                                  ;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
 8289  036322  013737  003176  172354          MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
 8290  036330  052737  000001  177572          BIS     #BIT0,@#SR0     ;TURN ON MEMORY MANAGEMENT
 8291  036336  042701  160000                  BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
 8292  036342  052701  140000                  BIS     #140000,R1
 8293  036346  010003                  22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8294  036350  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8295  036352  012321                  24$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8296  036354  032701  020000                  BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
 8297  036360  001405                          BEQ     26$             ;BR IF NO OVERFLOW
 8298  036362  062737  000200  172354          ADD     #200,@#KIPAR6   ;INCREMENT PAR BY 4K FOR NEW PAGE
 8299  036370  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 8300  036374  005302                  26$:    DEC     R2              ;DECREMENT WORD COUNTER
 8301  036376  001403                          BEQ     28$             ;BR IF ALL DONE
```

```
 8302  036400  005304                        DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8303  036402  001363                        BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
 8304  036404  000760                        BR      22$             ;BR TO REPEAT THE PATTERN
 8305  036406  042737  000001  177572  28$:  BIC     #BIT0,@#SR0     ;DISABLE MEMORY MANAGEMENT
 8306  036414  000410                        BR      44$             ;GO TO RETURN
 8307                                  ;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE
 8308  036416  010003                  30$:  MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8309  036420  011604                        MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8310  036422  012321                  34$:  MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8311  036424  005302                        DEC     R2              ;DECREMENT WORD COUNTER
 8312  036426  001403                        BEQ     44$             ;BR IF ALL DONE
 8313  036430  005304                        DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8314  036432  001373                        BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
 8315  036434  000770                        BR      30$             ;BR TO REPEAT THE PATTERN
 8316  036436  005726                  44$:  TST     (SP)+           ;POP THE STACK
 8317  036440  104410                        RESREG                  ;RESTORE R0-R5
 8318  036442  000207                        RTS     PC              ;RETURN
 8319
 8320
 8321
 8322                                  ;;*****************************************************************
 8323                                  .SBTTL  CMPBUF - SOFTWARE COMPARE DATA
 8324                                  ;*THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
 8325                                  ;*TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
 8326                                  ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
 8327                                  ;*00-15 (QUICK VERIFY DEFAULT DAT/ TEST).
 8328                                  ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
 8329                                  ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
 8330                                  ;;*****************************************************************
 8331  036444  104407                  CMPBUF: SAVREG                  ;SAVE R0-R5
 8332  036446  112737  000040  064154          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
 8333  036454  012737  000007  065530          MOV     #7,DF25+2
 8334  036462  013702  005564                  MOV     WDSXFR,R2       ;SET NO. OF WORDS
 8335  036466  005037  005534                  CLR     SCRACH          ;CLEAR COMPARE ERROR COUNT
 8336  036472  004737  043676                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
 8337  036476  005737  005536                  TST     PATRN           ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
 8338  036502  001007                          BNE     3$              ;BR IF NOT
 8339  036504  012700  006036                  MOV     #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
 8340  036510  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
 8341  036514  012701  065636                  MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
 8342  036520  000466                          BR      30$             ;PROCEED
 8343  036522  005000                  3$:     CLR     R0              ;INIT PATTERN NO.
 8344  036524  032701  000001          4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
 8345  036530  001003                          BNE     6$              ;BR IF THIS BIT IS SET
 8346  036532  005200                          INC     R0              ;INCREMENT PATTERN NUMBER
 8347  036534  006201                          ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
 8348  036536  000772                          BR      4$              ;BR TO CHECK NEXT BIT
 8349  036540  006300                  6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
 8350  036542  006300                          ASL     R0
 8351  036544  006300                          ASL     R0
 8352  036546  006300                          ASL     R0
 8353  036550  006300                          ASL     R0
 8354  036552  062700  006036                  ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
 8355  036556  012746  000020                  MOV     #16.,-(SP)      ;PATTERN WORD COUNT
 8356  036562  013701  005604                  MOV     PMA,R1          ;SET BUFFER ADDRESS
 8357  036566  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
```

```
8358  036572  100041                          BPL     30$             ;BR IF NOT PRESENT
8359                                  ;COMPARE LOOP FOR MEM MGT STARTS HERE
8360  036574  013737  003176  172354          MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
8361  036602  052737  000001  177572          BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
8362  036610  042701  160000                  BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
8363  036614  052701  140000                  BIS     #140000,R1
8364  036620  010003                  22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
8365  036622  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8366  036624  022321                  24$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
8367  036626  001404                          BEQ     25$             ;BR IF NO ERROR
8368  036630  013737  172354  001216          MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
8369  036636  000432                          BR      35$             ;GO HANDLE ERROR
8370  036640  032701  020000          25$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
8371  036644  001405                          BEQ     26$             ;BR IF NO OVERFLOW
8372  036646  062737  000200  172354          ADD     #200,@#KIPAR6   ;INCR PAR BY 4K FOR NEW PAGE
8373  036654  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
8374  036660  005302                  26$:    DEC     R2              ;DECREMENT WORD COUNTER
8375  036662  001002                          BNE     27$             ;BR IF NOT ALL DONE YET
8376  036664  000137  037330                  JMP     54$             ;ALL DONE - GET OUT
8377  036670  005304                  27$:    DEC     R4              ;DECREMENT PATTERN WORD COUNT
8378  036672  001354                          BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
8379  036674  000751                          BR      22$             ;BR TO REPEAT THE PATTERN
8380                                  ;COMPARE LOOP FOR NO MEM MGT STARTS HERE
8381  036676  010003                  30$:    MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
8382  036700  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8383  036702  022321                  34$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
8384  036704  001002                          BNE     31$             ;BR IF COMPARE ERROR
8385  036706  000137  037310                  JMP     40$             ;JUMP IF DATA COMPARES OK
8386  036712  105037  064154          31$:    CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
8387  036716  012737  000005  065530          MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
8388                                  ;COMMON COMPARE ERROR HANDLER
8389  036724  010237  001202          35$:    MOV     R2,$REG10       ;GET WORD NO.
8390  036730  163737  005564  001202          SUB     WDSXFR,$REG10
8391  036736  013737  001202  005572          MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
8392  036744  004737  036040                  JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
8393  036750  104407                          SAVREG                  ;SAVE R0-R5
8394  036752  013700  005566                  MOV     FINCYL,R0       ;GET CYL
8395  036756  113701  005570                  MOVB    FINTRK,R1       ;GET TRACK
8396  036762  113702  005571                  MOVB    FINSEC,R2       ;GET SECTOR
8397  036766  004737  042404                  JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
8398  036772  104410                          RESREG                  ;RESTORE R0-R5
8399  036774  032737  001000  005474          BIT     #BADSEC,RECODE
8400  037002  001132                          BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
8401  037004  005737  005534                  TST     SCRACH          ;CHECK THE ERROR COUNT
8402  037010  001024                          BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
8403  037012  105737  003143                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
8404  037016  001411                          BEQ     46$             ;BR IF NOT
8405  037020  013737  005262  001174          MOV     CRMPH0,$REG5    ;GET CURRENT MAP REG 0
8406  037026  013737  005260  001176          MOV     CRMPL0,$REG6
8407  037034  104116                          ERROR   116             ;DATA MISCOMPARE (11/70)
8408  037036  104117                          ERROR   117
8409  037040  000401                          BR      48$
8410  037042  104034                  46$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
8411  037044  012737  177777  001174  48$:    MOV     #-1,$REG5       ;INIT CYL NO.
8412  037052  005037  001176                  CLR     $REG6
8413  037056  005037  001200                  CLR     $REG7
```

```
 8414  037062  005237  005534          36$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
 8415  037066  032777  000001  142044          BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
 8416  037074  001004                          BNE     38$             ;BR TO REPORT ALL ERRORS
 8417  037076  022737  000012  005534          CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
 8418  037104  002511                          BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
 8419  037106  023737  001174  005566  38$:    CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
 8420  037114  001010                          BNE     42$             ;BR IF YES
 8421  037116  123737  001176  005570          CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
 8422  037124  001004                          BNE     42$             ;BR IF YES
 8423  037126  123737  001200  005571          CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
 8424  037134  001412                          BEQ     44$             ;BR IF SAME PACK ADDRESS
 8425  037136  013737  005566  001174  42$:    MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
 8426  037144  113737  005570  001176          MOVB    FINTRK,$REG6
 8427  037152  113737  005571  001200          MOVB    FINSEC,$REG7
 8428  037160  104115                          ERROR   115             ;TYPE NEW PACK ADDRESS
 8429  037162  005437  001202          44$:    NEG     $REG10          ;GET WORD NO.
 8430  037166  016337  177776  001204          MOV     -2(R3),$REG11   ;GET GOOD DATA
 8431  037174  016137  177776  001206          MOV     -2(R1),$REG12   ;GET BAD DATA
 8432  037202  013737  001202  001212          MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
 8433  037210  005037  001210                  CLR     $REG13
 8434  037214  006137  001212                  ROL     $REG14
 8435  037220  006137  001210                  ROL     $REG13
 8436  037224  063737  005604  001212          ADD     PMA,$REG14
 8437  037232  005537  001210                  ADC     $REG13
 8438  037236  063737  005606  001210          ADD     PMA+2,$REG13
 8439  037244  010137  001214                  MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
 8440  037250  162737  000002  001214          SUB     #2,$REG15
 8441  037256  104063                          ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
 8442  037260  032777  000100  141652          BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
 8443  037266  001020                          BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
 8444  037270  005737  005536          50$:    TST     PATRN           ;SEE IF DEFAULT DATA TEST
 8445  037274  001405                          BEQ     40$             ;BR IF YES
 8446  037276  005737  057710                  TST     $KT11           ;SEE IF MEM MGT PRESENT
 8447  037302  100002                          BPL     40$             ;BR IF NOT PRESENT
 8448  037304  000137  036640                  JMP     25$             ;GO HANDLE MEM MGT
 8449  037310  005302                  40$:    DEC     R2              ;DECREMENT WORD COUNTER
 8450  037312  001406                          BEQ     54$             ;BR IF ALL DONE
 8451  037314  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8452  037316  001002                          BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
 8453  037320  000137  036676                  JMP     30$             ;JUMP TO REPEAT THE PATTERN
 8454  037324  000137  036702          53$:    JMP     34$
 8455  037330  005737  057710          54$:    TST     $KT11           ;SEE IF MEM MGT PRESENT
 8456  037334  100003                          BPL     56$             ;BR IF NOT PRESENT
 8457  037336  042737  000001  177572          BIC     #BIT0,@#SR0     ;DISABLE MEM MGT
 8458  037344  005726                  56$:    TST     (SP)+           ;POP THE STACK
 8459  037346  104410                          RESREG                  ;RESTORE R0-R5
 8460  037350  000207                          RTS     PC              ;RETURN
 8461
 8462
 8463
 8464                                  ;;*****************************************************************
 8465                                  ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (^C) OR (^Z) TTY INPUT.
 8466                                  ;*IF (^C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (^Z)
 8467                                  ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
 8468                                  ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
 8469                                  ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (^C) OR (^Z),
```

```
                                        ;*NO ACTION IS TAKEN.
                                        ;*  CALL -   JSR     PC,CTLOUT
                                        ;*    OR -   CKEXIT
                                        ;;*****************************************************************
037352  122737  000001  003120  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
037360  001077                          BNE     10$             ;BR IF NOT RUNNING TESTS
037362  005737  005522                  TST     INTCHR          ;SEE IF ANY TTY INPUT
037366  001474                          BEQ     10$             ;BR IF NO INPUT
037370  105737  003116                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
037374  001446                          BEQ     12$             ;BR IF YES
037376  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
037404  001033                          BNE     4$              ;BR IF NOT (^C)
037406  012700  014374                  MOV     #DRVTST,R0      ;SET RETURN ADDR = DRVTST
037412  105737  003133          2$:     TSTB    XOVLAD          ;SEE IF XXDP CURRENTLY OVERLAID
037416  001402                          BEQ     6$              ;BR IF NOT
037420  004737  032010                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP, IF NECESSARY
037424  000005                  6$:     RESET                   ;RESET ALL DEVICES
037426  005037  005522                  CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
037432  005037  001304                  CLR     $TIMES          ;CLEAR THE ITER. COUNT
037436  105037  003134                  CLRB    XDPSVD          ;CLEAR THE XXDP SAVED FLAG
037442  012737  044156  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADDRESS
037450  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK
037454  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
037462  004737  042510                  JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
037466  005037  001102                  CLR     $TSTNM          ;CLEAR THE TEST NO.
037472  000110                          JMP     @R0             ;EXIT FROM TESTS
037474  122737  000032  005522  4$:     CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
037502  001016                          BNE     7$              ;BR IF NOT (^Z)
037504  012700  015754                  MOV     #INPUTP,R0      ;SET RETURN ADDR = INPUTP
037510  000740                          BR      2$              ;TAKE EXIT
037512  122737  000003  005522  12$:    CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
037520  001007                          BNE     7$              ;BR IF NOT
037522  104401  007643                  TYPE    ,HLTRQD         ;TYPE "HALT REQUESTED"
037526  104401  007613                  TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN RDY"
037532  012700  046126                  MOV     #HLTPRG,R0      ;SET HALT ADDRESS
037536  000725                          BR      2$              ;TAKE EXIT
037540  122737  000007  005522  7$:     CMPB    #007,INTCHR     ;SEE IF (^G) TYPED
037546  001002                          BNE     8$              ;BR IF NOT (^G)
037550  004737  031066                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
037554  004737  030536          8$:     JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
037560  000207                  10$:    RTS     PC              ;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.
                                        ;;*****************************************************************
037562  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
                                        ;LOAD THE R/W BUFFER WITH DATA
037564  012700  065636                  MOV     #RWBUF,R0       ;BUFFER ADDRESS
037570  012701  000400                  MOV     #400,R1         ;400(OCT) WORDS
037574  010320          4$:     MOV     R3,(R0)+        ;LOAD A BUFFER WORD
037576  005301                  DEC     R1              ;DECR COUNTER
```

```
                        BNE     4$              ;BR IF NOT DONE YET
                ;SET UP PARAMETERS
                        MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
                        MOV     #-400,P.WC(R5)  ;SET WORD COUNT
                ;WRITE THE DATA
                        MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
                        BIT     #BIT1,@SWR      ;SEE IF WRITES INHIBITED
                        BNE     6$              ;BR IF YES
                        JSR     PC,DRVCAL       ;WRITE THE DATA
                ;PERFORM WRITE CHECK
                6$:     MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
                        JSR     PC,DRVCAL       ;PERFORM WRITE CHECK
                        RESREG                  ;RESTORE R0-R5
                        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.
                ;;*******************************************************************
                LDMEM1: SAVREG                  ;SAVE R0-R5
                        CLR     R4              ;R4=0 INDICATES LDMEM1
                        MOV     #1,R3           ;INIT DATA WORD TO 1
                        BR      LDM0            ;PROCEED
                LDMEM2: SAVREG                  ;SAVE R0-R5
                        MOV     #-1,R4          ;R4=177777 INDICATES LDMEM2
                LDM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
                        MOV     PMA,R1          ;GET MEM ADRS
                        TST     $KT11           ;SEE IF MEM MGT
                        BPL     6$              ;BR IF NOT
                        BIC     #160000,R1      ;FORCE RELOCATION THRU PAR6
                        BIS     #140000,R1
                        MOV     SAVPAR,@#KIPAR6 ;INIT PAR6
                        BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
                6$:     MOV     R3,(R1)+        ;LOAD A WORD INTO BUFFER
                        TST     R4              ;SEE WHICH DATA DESIRED
                        BNE     8$              ;BR IF NOT INCREMENTING DATA
                        INC     R3              ;INCREMENT THE DATA
                8$:     TST     $KT11           ;SEE IF MEM MGT
                        BPL     10$             ;BR IF NOT
                        BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
                        BEQ     10$             ;BR IF NO OVERFLOW
                        ADD     #200,@#KIPAR6   ;INCR PAR BY 4K FOR NEW PAGE
                        BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
                10$:    DEC     R2              ;DECREMENT COUNTER
                        BNE     6$              ;BR IF NOT DONE LOADING YET
                        TST     $KT11           ;SEE IF MEM MGT
                        BPL     12$             ;BR IF NOT
                        BIC     #BIT0,@#SR0     ;TURN OFF MEM MGT
                12$:    RESREG                  ;RESTORE R0-R5
                        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.
                        ;;*****************************************************************
040022  104407          CKMEM1: SAVREG                  ;SAVE R0-R5
040024  005004                  CLR     R4              ;R4=0 INDICATES CKMEM1
040026  012703  000001          MOV     #1,R3           ;INIT DATA TO 1
040032  000403                  BR      CKM0            ;PROCEED
040034  104407          CKMEM2: SAVREG                  ;SAVE R0-R5
040036  012704  177777          MOV     #-1,R4          ;R4=177777 INDICATES CKMEM2
040042  013702  005564  CKM0:   MOV     WDSXFR,R2       ;GET NO. OF WORDS
040046  013701  005604          MOV     PMA,R1          ;GET MEM ADRS
040052  005037  005534          CLR     SCRACH
040056  112737  000040  064154  MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
040064  012737  000007  065530  MOV     #7,DF25+2
040072  004737  043676          JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
040076  005737  057710          TST     $KT11           ;SEE IF MEM MGT
040102  100012                  BPL     6$              ;BR IF NOT
040104  042701  160000          BIC     #160000,R1      ;FORCE RELOCATION THRU PAR6
040110  052701  140000          BIS     #140000,R1
040114  013737  003176  172354  MOV     SAVPAR,@#KIPAR6 ;INIT PAR6
040122  052737  000001  177572  BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
040130  020321          6$:     CMP     R3,(R1)+        ;COMPARE A DATA WORD
040132  001575                  BEQ     16$             ;BR IF DATA COMPARES OK
                        ;COMPARE ERROR HANDLER
040134  010237  001202          MOV     R2,$REG10       ;GET WORD NO.
040140  163737  005564  001202  SUB     WDSXFR,$REG10
040146  013737  001202  005572  MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
040154  004737  036040          JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
040160  104407                  SAVREG                  ;SAVE R0-R5
040162  013700  005566          MOV     FINCYL,R0       ;GET CYL
040166  113701  005570          MOVB    FINTRK,R1       ;GET TRACK
040172  113702  005571          MOVB    FINSEC,R2       ;GET SECTOR
040176  004737  042404          JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
040202  104410                  RESREG                  ;RESTORE R0-R5
040204  032737  001000  005474  BIT     #BADSEC,RECODE
040212  001145                  BNE     16$             ;BR IF LISTED- DON'T REPORT ERROR
040214  005737  057710          TST     $KT11           ;SEE IF MEM MGT
040220  100406                  BMI     8$              ;BR IF YES
040222  105037  064154          CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
040226  012737  000005  065530  MOV     #5,DF25+2       ;ADJ. ERROR DATA WORD COUNT
040234  000403                  BR      9$
040236  013737  172354  001216  8$:     MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
040244  005737  005534          9$:     TST     SCRACH          ;SEE IF FIRST ERROR IN THIS BLOCK
040250  001024                  BNE     10$             ;BR IF NOT FIRST ERROR
040252  105737  003143          TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
040256  001411                  BEQ     17$             ;BR IF NOT
040260  013737  005262  001174  MOV     CRMPH0,$REG5    ;CURRENT UB MAP REG 0
040266  013737  005260  001176  MOV     CRMPL0,$REG6
040274  104116                  ERROR   116             ;DATA MISCOMPARE (11/70)
040276  104117                  ERROR   117
```

```
8638  040300  000401                        BR      11$
8639  040302  104034                17$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
8640  040304  012737  177777  001174  11$:  MOV     #-1,$REG5       ;INIT CYL NO.
8641  040312  005037  001176                CLR     $REG6
8642  040316  005037  001200                CLR     $REG7
8643  040322  005237  005534          10$:  INC     SCRACH          ;INCREMENT THE ERROR COUNT
8644  040326  032777  000001  140604        BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
8645  040334  001004                        BNE     12$             ;BR TO REPORT ALL ERRORS
8646  040336  022737  000012  005534        CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
8647  040344  002512                        BLT     21$             ;BR IF ERROR LIMIT EXCEEDED
8648  040346  023737  001174  005566  12$:  CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
8649  040354  001010                        BNE     14$             ;BR IF YES
8650  040356  123737  001176  005570        CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
8651  040364  001004                        BNE     14$             ;BR IF YES
8652  040366  123737  001200  005571        CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
8653  040374  001412                        BEQ     15$             ;BR IF SAME PACK ADDRESS
8654  040376  013737  005566  001174  14$:  MOV     FINCYL,$REG5    ;SET NEW PACK ADR FOR PRINTOUT
8655  040404  113737  005570  001176        MOVB    FINTRK,$REG6
8656  040412  113737  005571  001200        MOVB    FINSEC,$REG7
8657  040420  104115                        ERROR   115             ;TYPE NEW PACK ADRS
8658  040422  005437  001202          15$:  NEG     $REG10          ;GET WORD NO.
8659  040426  010337  001204                MOV     R3,$REG11       ;GOOD DATA
8660  040432  016137  177776  001206        MOV     -2(R1),$REG12   ;BAD DATA
8661  040440  013737  001202  001212        MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
8662  040446  005037  001210                CLR     $REG13
8663  040452  006137  001212                ROL     $REG14
8664  040456  006137  001210                ROL     $REG13
8665  040462  063737  005604  001212        ADD     PMA,$REG14
8666  040470  005537  001210                ADC     $REG13
8667  040474  063737  005606  001210        ADD     PMA+2,$REG13
8668  040502  010137  001214                MOV     R1,$REG15       ;GET VIRT ADRS FOR PRINTOUT
8669  040506  162737  000002  001214        SUB     #2,$REG15
8670  040514  104063                        ERROR   63              ;TYPE GOOD AND BAD DATA
8671  040516  032777  000100  140414        BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8672  040524  001022                        BNE     21$             ;BR IF JUST 1
8673                          ;PROCEED WITH COMPARISONS
8674  040526  005704                16$:    TST     R4              ;SEE WHICH DATA DESIRED
8675  040530  001001                        BNE     18$             ;BR IF NOT INCREASING DATA
8676  040532  005203                        INC     R3              ;INCREMENT THE DATA WORD
8677  040534  005737  057710          18$:  TST     $KT11           ;SEE IF MEM MGT
8678  040540  100010                        BPL     20$             ;BR IF NOT
8679  040542  032701  020000                BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
8680  040546  001405                        BEQ     20$             ;BR IF NO OVERFLOW
8681  040550  062737  000200  172354        ADD     #200,@#KIPAR6   ;INCR. PAR BY 4K FOR NEW PAGE
8682  040556  042701  020000                BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
8683  040562  005302                20$:    DEC     R2              ;DECR COUNTER
8684  040564  001402                        BEQ     21$             ;BR IF DONE COMPARING
8685  040566  000137  040130                JMP     6$              ;CONTINUE COMPARING WORDS
8686  040572  005737  057710          21$:  TST     $KT11           ;SEE IF MEM MGT PRESENT
8687  040576  100003                        BPL     22$             ;BR IF NOT
8688  040600  042737  000001  177572        BIC     #BIT0,@#SR0     ;TURN OFF MEM MGT
8689  040606  104410                22$:    RESREG                  ;RESTORE R0-R5
8690  040610  000207                        RTS     PC              ;RETURN
8691
8692
8693
```

```
                                        ;;*****************************************************************
                                        ;*SETUPM - SUBROUTINE TO INIT PARAMS FOR NPR/MEMORY TESTS.
                                        ;;*****************************************************************
040612  013765  005660  000002  SETUPM: MOV     FC,P.CYLN(R5)    ;SET CYL
040620  113765  005506  000004           MOVB    FS,P.SECT(R5)    ;SET SECTOR
040626  113765  005664  000005           MOVB    FT,P.TRCK(R5)    ;SET TRACK
040634  023737  005660  017776           CMP     FC,LCM6          ;SEE IF CYL SHOULD BE SCALED DOWN
040642  003403                           BLE     2$               ;BR IF NOT
040644  013765  017776  000002           MOV     LCM6,P.CYLN(R5)  ;SCALE DOWN THE CYLINDER
040652  012737  065636  005600  2$:      MOV     #RWBUF,MA        ;GET MEMORY ADDRESS
040660  005737  000170                   TST     KILLDR           ;SEE IF LOADER CAN BE OVERLAYED
040664  001003                           BNE     4$               ;BR IF YES
040666  062737  006000  005600           ADD     #6000,MA         ;LEAVE ROOM TO SAVE LOADER
040674  042737  003777  005600  4$:      BIC     #3777,MA
040702  062737  004000  005600           ADD     #4000,MA
040710  005037  005602                   CLR     MA+2
040714  000207                           RTS     PC               ;RETURN


                                        ;;*****************************************************************
                                        ;* REFNEM - THIS SUBROUTINE PERFORMS A SUBSYSTEM COMMAND, WHILE
                                        ;*REPEATEDLY REFERENCING A NON-EXISTENT MEMORY LOCATION (760000). THIS
                                        ;*IS DONE FOR THE PURPOSE OF CAUSING UNIBUS CONTENTION DURING DISK
                                        ;*NPR'S, SINCE THE BUS IS SIEZED FOR 5-20 USEC AT EACH NEM TIME-OUT.
                                        ;*ALL DRIVER PARAMETERS (INCLUDING THE COMMAND) MUST BE SET IN THE
                                        ;*PARAMETER BLOCK ON ENTRY, AND R3 MUST CONTAIN THE TIMING CONSTANT
                                        ;*WHICH REGULATES THE FREQUENCY OF NEM REFERENCES, WHILE WAITING FOR
                                        ;*COMMAND COMPLETION.
                                        ;;*****************************************************************
040716                                  REFNEM:
040716  004737  037352                   JSR     PC,CTLOUT        ;CHECK FOR (^C) OR (^Z) KBD INPUT
040722  105037  003123                   CLRB    DONE             ;CLEAR THE DONE FLAG
040726  004737  042570                   JSR     PC,STRCMD        ;STORE PREV AND CURRENT COMMANDS
040732  010537  040750                   MOV     R5,4$            ;SET PARAM BLK ADDRESS
040736  012737  041004  000004           MOV     #20$,@#ERRVEC    ;SET TIME-OUT VECTOR ADDRESS
040744  004737  052562                   JSR     PC,C.INIT        ;CALL DRIVER
040750  000000                  4$:      .WORD
040752  004737  047132          6$:      JSR     PC,W.WTCH        ;CALL WATCH DOG
040756  105737  003123                   TSTB    DONE             ;DONE SET ?
040762  001012                           BNE     26$              ;BR IF YES
040764  010304                           MOV     R3,R4            ;GET COPY OF R3
040766  001402                           BEQ     12$              ;BR IF TIMER = 0
040770  005304                  10$:     DEC     R4               ;DECREMENT TIMER
040772  001376                           BNE     10$              ;BR IF NOT DONE TIMING
040774  005737  160000          12$:     TST     @#160000         ;REFERENCE NON-EXISTENT MEMORY
041000  104111                           ERROR   111              ;"NO NEM WHEN EXPECTED"
041002  000401                           BR      22$
041004  022626                  20$:     CMP     (SP)+,(SP)+      ;RESTORE THE STACK
041006  000761                  22$:     BR      6$               ;KEEP WAITING FOR COMMAND COMPLETION
041010  012737  000006  000004  26$:     MOV     #6,@#ERRVEC      ;RESTORE TIME-OUT VECTOR
041016  000207                           RTS     PC               ;RETURN


                                        ;;*****************************************************************
                                        ;*INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
```

```
                                ;;*****************************************************************
041020  104407                  INCRMA: SAVREG                  ;SAVE R0-R5
041022  005001                          CLR     R1
041024  016500  000012                  MOV     P.WC(R5),R0     ;GET WORD COUNT
041030  001002                          BNE     4$              ;BR IF NOT 65,536(DEC)
041032  005201                          INC     R1              ;SET HI BIT
041034  000401                          BR      6$              ;CONTINUE
041036  005400                  4$:     NEG     R0              ;GET TRUE WORD COUNT
041040  000241                  6$:     CLC                     ;DOUBLE THE WORD COUNT TO GET BYTES
041042  006100                          ROL     R0
041044  006101                          ROL     R1
041046  060037  005600                  ADD     R0,MA           ;ADD IT TO COMPUTE NEW MA
041052  005537  005602                  ADC     MA+2
041056  060137  005602                  ADD     R1,MA+2
041062  104410                          RESREG                  ;RESTORE R0-R5
041064  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
                                ;*INTO BSSOFT.
                                ;;*****************************************************************
041066                          REDBSF:
041066  004737  032250                  JSR     PC,INITSS       ;INIT THE S.S.
041072  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FMT
041100  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
041106  013765  020000  000002          MOV     LSTCYL,P.CYLN(R5)       ;SET CYL = 632/1456
041114  112765  000002  000005          MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
041122  012703  000012                  MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
041126  105737  003125                  TSTB    FORMAT
041132  001403                          BEQ     6$              ;BR IF 22 SECTORS
041134  105265  000004                  INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
041140  005203                          INC     R3
041142  012765  177400  000012  6$:     MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
041150  012765  004224  000010          MOV     #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
041156  012737  041332  003046  8$:     MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
041164  105037  003140                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
041170  004737  042510                  JSR     PC,DRVCAL       ;READ THE FACTORY BSF
041174  105737  003140                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
041200  001413                          BEQ     12$             ;BR IF NOT
041202  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
041210  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
041214  001360                          BNE     8$              ;BR IF NOT YET
041216  004737  043676          10$:    JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
041222  104120                          ERROR   120             ;ABORTING- BAD BSF READ
041224  000137  046126                  JMP     HLTPRG          ;HALT- CAN'T PROCEED
041230  012765  003224  000010  12$:    MOV     #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
041236  110365  000004                  MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
041242  012703  000026                  MOV     #22.,R3         ;SET 22 SECTOR LIMIT
041246  105737  003125                  TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
041252  001402                          BEQ     14$             ;BR IF YES
041254  012703  000023                  MOV     #19.,R3         ;SET 20 SECTOR LIMIT
041260  012737  041332  003046  14$:    MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
041266  105037  003140                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
041272  004737  042510                  JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
```

```
 8806  041276  105737  003140                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
 8807  041302  001407                          BEQ     16$             ;BR IF NOT
 8808  041304  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
 8809  041312  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
 8810  041316  003760                          BLE     14$             ;BR IF NOT YET
 8811  041320  000736                          BR      10$             ;REPORT ERROR AND ABORT
 8812  041322  012737  044156  003046  16$:    MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
 8813  041330  000207                          RTS     PC              ;RETURN
 8814
 8815                                  ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
 8816                                  ;*POSSIBLE ERRORS IN READING BAD SECTORS.
 8817  041332  032765  130600  000034  ENDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
 8818  041340  001002                          BNE     4$              ;BR IF A READ ERROR OCCURRED
 8819  041342  000137  044156                  JMP     ERRHDL          ;GO HANDLE OTHER ERROR
 8820  041346  105237  003140          4$:     INCB    WCEFLG          ;SET ERROR FLAG
 8821  041352  000137  046334                  JMP     RETNML          ;TAKE NORMAL DRIVER RETURN
 8822
 8823
 8824
 8825                                  ;;*****************************************************************
 8826                                  ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
 8827                                  ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
 8828                                  ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
 8829                                  ;*2 BSF'S.
 8830                                  ;;*****************************************************************
 8831  041356  104407                  TYPBSF: SAVREG                  ;SAVE R0-R5
 8832  041360  005004                          CLR     R4              ;INIT BSF FILE INDICATOR
 8833  041362  012700  004234                  MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
 8834  041366  104401  011123                  TYPE    ,FACTBS         ;TYPE "FACTORY"
 8835  041372  104401  011150                  TYPE    ,BDSECT         ;TYPE "BAD SECTORS"
 8836  041376  104401  063721          3$:     TYPE    ,DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
 8837  041402  104401  001315                  TYPE    ,$CRLF
 8838  041406  005001                          CLR     R1              ;INIT LIMIT COUNTER
 8839  041410  005710                  4$:     TST     (R0)            ;SEE IF A SECTOR LISTED HERE
 8840  041412  100007                          BPL     8$              ;BR IF YES
 8841  041414  005701                          TST     R1              ;SEE IF FILE IS EMPTY
 8842  041416  001032                          BNE     14$             ;BR IF NOT EMPTY
 8843  041420  104401  011170                  TYPE    ,NOFALS         ;TYPE "NONE"
 8844  041424  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 8845  041430  000425                          BR      14$
 8846  041432  012046                  8$:     MOV     (R0)+,-(SP)     ;GET A CYL NO.
 8847  041434  104402                          TYPOC                   ;TYPE IT
 8848  041436  104401  013252                  TYPE    ,SPACE2         ;TYPE SEPARATORS
 8849  041442  011003                          MOV     (R0),R3         ;GET TRACK AND SECTOR
 8850  041444  105003                          CLRB    R3
 8851  041446  000303                          SWAB    R3              ;GET THE TRACK NO.
 8852  041450  010346                          MOV     R3,-(SP)
 8853  041452  104402                          TYPOC                   ;TYPE IT
 8854  041454  104401  013252                  TYPE    ,SPACE2         ;TYPE SEPARATORS
 8855  041460  111003                          MOVB    (R0),R3         ;GET SECTOR NO.
 8856  041462  010346                          MOV     R3,-(SP)
 8857  041464  104402                          TYPOC                   ;TYPE IT
 8858  041466  104401  001315                  TYPE    ,$CRLF
 8859  041472  005720                          TST     (R0)+           ;INCR THE POINTER
 8860  041474  005201                          INC     R1              ;INCR THE COUNTER
 8861  041476  020127  000176                  CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
```

```
 8862  041502  002742                    BLT     4$              ;BR IF NOT YET
 8863  041504  005704            14$:    TST     R4              ;SEE IF BOTH FILES TYPED YET
 8864  041506  001006                    BNE     18$             ;BR IF YES
 8865  041510  005204                    INC     R4              ;SET INDICATOR FOR SOFT. FILE
 8866  041512  012700  003234            MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
 8867  041516  104401  011136            TYPE    ,SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
 8868  041522  000725                    BR      3$              ;GO TYPE SOFT. BSF
 8869  041524  104410            18$:    RESREG                  ;RESTORE R0-R5
 8870  041526  000207                    RTS     PC              ;RETURN
 8871
 8872
 8873
 8874                            ;;*****************************************************************
 8875                            ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
 8876                            ;*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
 8877                            ;;*****************************************************************
 8878  041530  104407            FINMEM: SAVREG                  ;SAVE R0-R5
 8879  041532  016500  000002            MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
 8880  041536  116501  000005            MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
 8881  041542  116502  000004            MOVB    P.SECT(R5),R2   ;SECTOR NO.
 8882  041546  005003                    CLR     R3              ;INIT SECTOR COUNT TO 0
 8883  041550  026500  000030    6$:     CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
 8884  041554  001006                    BNE     8$              ;BR IF NOT EQUAL
 8885  041556  126501  000027            CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
 8886  041562  001003                    BNE     8$              ;BR IF NOT EQUAL
 8887  041564  126502  000026            CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
 8888  041570  001404                    BEQ     12$             ;BR IF ADRS ARE EQUAL
 8889  041572  005203            8$:     INC     R3              ;INCR SECTOR COUNT
 8890  041574  004737  041614            JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
 8891  041600  000763                    BR      6$              ;GO COMPARE ADDRESSES
 8892  041602  000303            12$:    SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
 8893  041604  010337  005564            MOV     R3,WDSXFR       ;STORE IT
 8894  041610  104410                    RESREG                  ;RESTORE R0-R5
 8895  041612  000207                    RTS     PC              ;RETURN
 8896
 8897
 8898
 8899                            ;;*****************************************************************
 8900                            ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
 8901                            ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
 8902                            ;*IN R2.
 8903                            ;;*****************************************************************
 8904  041614  012704  000025    INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
 8905  041620  105737  003125            TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
 8906  041624  001402                    BEQ     4$              ;BR IF YES
 8907  041626  012704  000023            MOV     #19.,R4         ;SET SECTOR LIMIT
 8908  041632  005202            4$:     INC     R2              ;INCR. SECTOR NO.
 8909  041634  020204                    CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
 8910  041636  003407                    BLE     6$              ;BR IF NOT
 8911  041640  005002                    CLR     R2              ;SET SECTOR = 0
 8912  041642  005201                    INC     R1              ;INCR. TRACK NO.
 8913  041644  020127  000002            CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
 8914  041650  003402                    BLE     6$              ;BR IF NOT
 8915  041652  005001                    CLR     R1              ;SET TRACK = 0
 8916  041654  005200                    INC     R0              ;INCR. CYLINDER NO.
 8917  041656  000207            6$:     RTS     PC              ;RETURN
```

```
                                ;;*****************************************************************
                                ;*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
                                ;*THIS SUBROUTINE UPDATES PMA,PMA+2,MEM MGT REGS,UNIBUS MAP,P.BALO(R5),
                                ;*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
                                ;*TERMINATED BY A BAD SECTOR ERROR (BSE).
                                ;;*****************************************************************
041660  104407                  MIDXFR: SAVREG                  ;SAVE R0-R5
041662  004737  041530                  JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
041666  016500  000030                  MOV     P.DCYL(R5),R0   ;GET CYL NO.
041672  116501  000027                  MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
041676  116502  000026                  MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
041702  004737  041614                  JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
041706  010065  000002                  MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
041712  110165  000005                  MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
041716  110265  000004                  MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
041722  013703  005564                  MOV     WDSXFR,R3       ;GET NO. OF WORDS XFERRED
041726  062703  000400                  ADD     #400,R3         ;SKIP BAD SECTOR
041732  060365  000012                  ADD     R3,P.WC(R5)     ;UPDATE P.WC(R5)
041736  005004                          CLR     R4              ;GET BYTES XFERRED
041740  006103                          ROL     R3
041742  006104                          ROL     R4
041744  060337  005604                  ADD     R3,PMA          ;UPDATE PMA,PMA+2
041750  005537  005606                  ADC     PMA+2
041754  060437  005606                  ADD     R4,PMA+2
041760  013765  005604  000010          MOV     PMA,P.BALO(R5)  ;UPDATE P.BALO(R5)
041766  013700  005606                  MOV     PMA+2,R0
041772  042700  177774                  BIC     #177774,R0
041776  150065  000007                  BISB    R0,P.BAHI(R5)   ;UPDATE P.BAHI(R5)
042002  005737  057710                  TST     $KT11           ;SEE IF MEM MGT
042006  100002                          BPL     16$
042010  004737  031606                  JSR     PC,PREPAR       ;UPDATE MEM MGT AND UNIBUS MAP
042014  104410                  16$:    RESREG                  ;RESTORE R0-R5
042016  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
                                ;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
                                ;;*****************************************************************
042020  104407                  SVPRMS: SAVREG                  ;SAVE R0-R5
042022  012700  002632                  MOV     #PARM0+2,R0     ;ADRS OF PARAMS
042026  012701  005552                  MOV     #SAVPRS,R1      ;ADRS OF SAVE AREA
042032  016537  000012  005564          MOV     P.WC(R5),WDSXFR  ;GET WORD COUNT
042040  005437  005564                  NEG     WDSXFR          ;MAKE IT POSITIVE
042044  012737  065636  005604          MOV     #RWBUF,PMA      ;INIT PMA TO RWBUF
042052  005037  005606                  CLR     PMA+2           ;INIT PMA+2 TO 0
042056  005737  005536                  TST     PATRN           ;SEE IF DEFAULT DATA TEST
042062  001414                          BEQ     GTO             ;BR IF YES
042064  013737  005600  005604          MOV     MA,PMA          ;SET PMA=MA
042072  013737  005602  005606          MOV     MA+2,PMA+2      ;SET PMA+2=MA+2
042100  000405                          BR      GTO
042102  104407                  GTPRMS: SAVREG                  ;SAVE R0-R5
```

```
 8974  042104  012700  005552            MOV     #SAVPRS,R0      ;ADRS OF SAVE AREA
 8975  042110  012701  002632            MOV     #PARM0+2,R1     ;ADRS OF PARAMS
 8976  042114  012702  000005    GTO:    MOV     #5,R2           ;SET FOR 5 WORDS
 8977  042120  012021            6$:     MOV     (R0)+,(R1)+     ;MOVE A WORD
 8978  042122  005302                    DEC     R2              ;SEE IF DONE YET
 8979  042124  001375                    BNE     6$              ;BR IF NOT YET
 8980  042126  104410                    RESREG                  ;RESTORE R0-R5
 8981  042130  000207                    RTS     PC              ;RETURN
 8982
 8983
 8984
 8985                            ;;*****************************************************************
 8986                            ;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS, IF
 8987                            ;*ENCOUNTERED. THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
 8988                            ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED.
 8989                            ;;*****************************************************************
 8990  042132  010446            TRNSFR: MOV     R4,-(SP)        ;SAVE R4 ON STACK
 8991  042134  005004                    CLR     R4              ;CLEAR BSE ERROR INDICATOR
 8992  042136  105037  003140            CLRB    WCEFLG          ;INIT WCE ERROR FLAG TO 0
 8993  042142  004737  042510    4$:     JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
 8994  042146  032737  000100  005474    BIT     #WCERR,RECODE   ;SEE IF WCE ERROR OCCURRED
 8995  042154  001403                    BEQ     5$              ;BR IF NOT
 8996  042156  112737  000001  003140    MOVB    #1,WCEFLG       ;SET WCE ERROR FLAG
 8997  042164  032737  000002  005474    5$:     BIT     #BSERR,RECODE   ;SEE IF BSE ERROR OCCURRED
 8998  042172  001406                    BEQ     6$              ;BR IF NOT
 8999  042174  005204                    INC     R4              ;INCR BSE ERROR INDICATOR
 9000  042176  004737  041660            JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
 9001  042202  005765  000012            TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
 9002  042206  001355                    BNE     4$              ;BR IF NOT
 9003  042210  005704            6$:     TST     R4              ;SEE IF ANY BSE ERRORS OCCURRED
 9004  042212  001411                    BEQ     8$              ;BR IF NOT
 9005  042214  004737  042102            JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
 9006  042220  004737  042020            JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
 9007  042224  005737  057710            TST     $KT11           ;SEE IF MEM MGT PRESENT
 9008  042230  100002                    BPL     8$              ;BR IF NOT
 9009  042232  004737  031606            JSR     PC,PREPAR       ;PREPARE MEM MGT AND U.M.
 9010  042236  012604            8$:     MOV     (SP)+,R4        ;RESTORE R4
 9011  042240  000207                    RTS     PC              ;RETURN
 9012
 9013
 9014
 9015                            ;;*****************************************************************
 9016                            .SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
 9017                            ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
 9018                            ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
 9019                            ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
 9020                            ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
 9021                            ;*
 9022                            ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
 9023                            ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
 9024                            ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
 9025                            ;;*****************************************************************
 9026  042242  010237  001266    SRHTBS: MOV     R2,$TMP2
 9027  042246  010337  001270            MOV     R3,$TMP3
 9028  042252  012637  001272            MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
 9029  042256  011402                    MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
```

```
9030  042260  012437  001264                 MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
9031  042264  062737  001000  001264         ADD     #1000,$TMP1
9032  042272  005003                         CLR     R3              ;CLEAR R3 FOR COUNT
9033  042274  062702  000010                 ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
9034  042300  005712                 1$:     TST     (R2)            ;TEST IF ALL ONES
9035  042302  100430                         BMI     5$              ;YES-DONE
9036  042304  020012                         CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
9037  042306  001406                         BEQ     3$              ;YES-GO CHECK TRACK
9038  042310  062702  000004                 ADD     #4,R2           ;ELSE BUMP POINTER
9039  042314  020237  001264                 CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
9040  042320  002021                         BGE     5$              ;YES-EXIT
9041  042322  000766                         BR      1$              ;LOOP
9042  042324  005722                 3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
9043  042326  011237  001302                 MOV     (R2),$TMP10     ;GET TRK/SEC WORD
9044  042332  042737  174377  001302         BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
9045  042340  123701  001303                 CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
9046  042344  001402                         BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
9047  042346  005722                         TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
9048  042350  000753                         BR      1$              ;GO TEST NEXT CYL
9049  042352  005203                 4$:     INC     R3              ;BUMP BAD SECTOR COUNT
9050  042354  012246                         MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
9051  042356  042716  177700                 BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
9052  042362  000746                         BR      1$              ;GO CHECK REST OF FILE
9053  042364  010346                 5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
9054  042366  013702  001266                 MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
9055  042372  013703  001270                 MOV     $TMP3,R3        ;REGISTERS
9056  042376  013746  001272                 MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
9057  042402  000204                         RTS     R4              ;RETURN
9058                                 ;;*****************************************************************
9059                                 .SBTTL  BAD SECTOR CHECK ROUTINE
9060                                 ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
9061                                 ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
9062                                 ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
9063                                 ;*IS NOT LISTED THE FLAG IS RESET.
9064                                 ;;*****************************************************************
9065  042404  104407                 BDSRCK: SAVREG
9066  042406  012737  004224  042420         MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
9067  042414  004437  042242         2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
9068  042420  000000                 1$:     .WORD                   ;TABLE ADDRESS GOES HERE
9069  042422  012603                         MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
9070  042424  001015                         BNE     6$              ;IF ANY, GO TEST WHICH ONES
9071  042426  023727  042420  003224 7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
9072  042434  001404                         BEQ     3$              ;IF YES-EXIT
9073  042436  012737  003224  042420         MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
9074  042444  000763                         BR      2$              ;GO SEARCH IT
9075  042446  042737  001000  005474 3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
9076  042454  104410                 4$:     RESREG
9077  042456  000207                         RTS     PC              ;RETURN
9078  042460  022602                 6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
9079  042462  001403                         BEQ     8$              ;YES-GO SET BIT & EXIT
9080  042464  005303                         DEC     R3              ;DECREMENT BAD SECTOR COUNT
9081  042466  001374                         BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
9082  042470  000756                         BR      7$              ;ELSE GO SEARCH OTHER TABLE
9083  042472  052737  001000  005474 8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
9084  042500  005303                 9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
9085  042502  001764                         BEQ     4$              ;NUMBER
```

```
 9086  042504  005726                    TST     (SP)+
 9087  042506  000774                    BR      9$              ;EXIT
 9088
 9089
 9090
 9091                                ;;*****************************************************************
 9092                                .SBTTL  CALL DRIVER ROUTINE
 9093                                ;*ENTRY JSR PC,DRVCAL
 9094                                ;*              WITH R5 POINTING TO PARAMETER BLOCK
 9095                                ;*RETURN RTS PC
 9096                                ;*
 9097                                ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
 9098                                ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
 9099                                ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
 9100                                ;*BLOCK WHEN THE ROUTINE IS CALLED.
 9101                                ;*
 9102                                ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
 9103                                ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
 9104                                ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
 9105                                ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
 9106                                ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
 9107                                ;;*****************************************************************
 9108  042510                        DRVCAL:
 9109  042510  104407                        SAVREG                  ;SAVE R0-R5
 9110  042512  004737  035552                JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
 9111  042516  004737  037352                JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
 9112  042522  105037  003123                CLRB    DONE            ;CLEAR DONE FLAG
 9113  042526  105037  003136                CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
 9114  042532  004737  042570                JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
 9115                                ;       BICB    #B.CDT,P.CS1H(R5) ;CLEAR THE RK07 DRIVE TYPE BIT
 9116                                ;       TSTB    TYPFMT          ;CURRENT DRIVE IS AN RK07 ?
 9117                                ;       BEQ     9$              ;BRANCH IF NOT
 9118                                ;       BISB    #B.CDT,P.CS1H(R5) ;SET THE RK07 DRIVE TYPE BIT
 9119  042536  000240                9$:     NOP                     ;
 9120  042540  010537  042550                MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
 9121  042544  004737  052562                JSR     PC,C.INIT       ;CALL DRIVER
 9122  042550  000000                4$:     .WORD                   ;P.B. ADDRESS GOES HERE
 9123  042552  004737  047132        6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
 9124  042556  105737  003123                TSTB    DONE            ;DONE SET?
 9125  042562  001775                        BEQ     6$              ;NO-LOOP
 9126  042564  104410                12$:    RESREG                  ;RESTORE R0-R5
 9127  042566  000207                        RTS     PC              ;YES-RETURN
 9128
 9129
 9130
 9131                                ;;*****************************************************************
 9132                                ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
 9133                                ;;*****************************************************************
 9134  042570  104407                STRCMD: SAVREG                  ;SAVE R0-R5
 9135  042572  012701  005224                MOV     #PRVCMD,R1      ;ADDR OF PREV COMMAND STORAGE
 9136  042576  012700  005240                MOV     #COMSTR,R0      ;ADDR OF CURRENT COMMAND STORAGE
 9137                                ;STORE PREVIOUS COMMAND
 9138  042602  012021                        MOV     (R0)+,(R1)+
 9139  042604  012021                        MOV     (R0)+,(R1)+
 9140  042606  012021                        MOV     (R0)+,(R1)+
 9141  042610  012021                        MOV     (R0)+,(R1)+
```

```
 9142  042612  012021                          MOV     (R0)+,(R1)+
 9143  042614  012021                          MOV     (R0)+,(R1)+
 9144  042616  105737  003143                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 9145  042622  001414                          BEQ     4$              ;BR IF NOT
 9146  042624  013737  005262  005256          MOV     CRMPH0,PRMPH0   ;STORE PREV U.B. MAP REG 0
 9147  042632  013737  005260  005254          MOV     CRMPL0,PRMPL0
 9148  042640  013737  170202  005262          MOV     @#MAPH00,CRMPH0 ;STORE CURRENT U.B. MAP REG 0
 9149  042646  013737  170200  005260          MOV     @#MAPL00,CRMPL0
 9150                                  ;STORE CURRENT COMMAND
 9151  042654  012701  005240          4$:     MOV     #COMSTR,R1
 9152  042660  012521                          MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BLK
 9153  042662  012521                          MOV     (R5)+,(R1)+
 9154  042664  012521                          MOV     (R5)+,(R1)+
 9155  042666  012521                          MOV     (R5)+,(R1)+
 9156  042670  012521                          MOV     (R5)+,(R1)+
 9157  042672  012521                          MOV     (R5)+,(R1)+
 9158  042674  104410                          RESREG                  ;RESTORE R0-R5
 9159  042676  000207                          RTS     PC              ;RETURN
 9160
 9161
 9162
 9163                                  ;;**********************************************************************
 9164                                  .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
 9165                                  ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
 9166                                  ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
 9167                                  ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
 9168                                  ;*ROUTINE.
 9169                                  ;;**********************************************************************
 9170  042700  152737  000377  003123  ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
 9171  042706  032737  100000  005474          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
 9172  042714  001403                          BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
 9173  042716  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 9174  042722  000413                          BR      1$              ;EXIT
 9175  042724  105737  003126          2$:     TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
 9176  042730  001410                          BEQ     1$              ;NO - SKIP TO EXIT
 9177  042732  005037  001174                  CLR     $REG5
 9178  042736  113737  003126  001174          MOVB    ERRCNT,$REG5    ;GET RETRY COUNT
 9179  042744  104101                          ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
 9180  042746  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 9181  042752  005037  005474          1$:     CLR     RECODE          ;CLEAR RECOVERY FLAGS
 9182  042756  000207                          RTS     PC              ;RETURN
 9183                                  ;;**********************************************************************
 9184                                  .SBTTL  TYPE ERROR ROUTINE
 9185                                  ;*ENTRY -       JSR     PC,TYPERR
 9186                                  ;*RETURN -      RTS     PC
 9187                                  ;*
 9188                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 9189                                  ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
 9190                                  ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
 9191                                  ;*THE ERROR.
 9192                                  ;;**********************************************************************
 9193  042760  104407                  TYPERR: SAVREG
 9194  042762  105237  003130                  INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
 9195  042766  032737  000040  005702          BIT     #BIT5,CS        ;SEE IF DRIVE SHOULD BE DROPPED
 9196                                                                  ;  IF ERROR LIMIT EXCEEDED
 9197  042774  001412                          BEQ     9$              ;BR IF NOT
```

```
 9198  042776  123727  003130  000024          CMPB    DRVERS,#20.     ;SEE IF 20(DEC) ERRORS EXCEEDED
 9199  043004  002406                          BLT     9$              ;BR IF NOT
 9200  043006  105037  003130                  CLRB    DRVERS          ;CLEAR DRIVE ERROR COUNT
 9201  043012  104401  011010                  TYPE    ,DROPDR         ;TYPE "DROPPING DRIVE"
 9202  043016  000137  017272                  JMP     NEWDRV          ;PROCEED TO TEST NEXT DRIVE
 9203  043022  032777  020000  136110  9$:     BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
 9204  043030  001402                          BEQ     6$              ;BR IF NO
 9205  043032  000137  043434                  JMP     20$
 9206  043036  005000                  6$:     CLR     R0              ;CLR R0 FOR ERROR NUMBER
 9207  043040  005005                          CLR     R5              ;INIT INDENT INDICATOR
 9208  043042  005105                          COM     R5
 9209  043044  113700  001114                  MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
 9210  043050  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
 9211  043052  006300                          ASL     R0
 9212  043054  006300                          ASL     R0
 9213  043056  006300                          ASL     R0
 9214  043060  062700  001400          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
 9215  043064  012037  043104                  MOV     (R0)+,2$        ;GET EM POINTER
 9216  043070  001406                          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 9217  043072  104401  013236                  TYPE    ,CR2LF
 9218  043076  104401  060346                  TYPE    ,AS2SP2         ;TYPE "** "
 9219  043102  104401                          TYPE                    ;TYPE ERROR MESSAGE (EM)
 9220  043104  000000                  2$:     .WORD   0               ;EM POINTER GOES HERE
 9221  043106  012037  043262          3$:     MOV     (R0)+,4$        ;GET DH POINTER
 9222  043112  001467                          BEQ     5$              ;BR IF THERE ISN'T ONE
 9223  043114  104401  001315                  TYPE    ,$CRLF
 9224  043120  104401  060334                  TYPE    ,TSTMSG         ;TYPE "    TEST "
 9225  043124  013746  001102                  MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
 9226  043130  104403                          TYPOS                   ;TYPE IT
 9227  043132     002                          .BYTE   2               ;2 DIGITS
 9228  043133     000                          .BYTE   0               ;SUPPRESS LEADING ZEROS
 9229  043134  104401  013236                  TYPE    ,CR2LF
 9230  043140  032777  010000  135772          BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 9231  043146  001133                          BNE     20$             ;BR IF YES
 9232  043150  104401  063066                  TYPE    ,DH105          ;TYPE "PREVIOUS COMMAND :"
 9233  043154  104401  001315                  TYPE    ,$CRLF
 9234  043160  104401  063231                  TYPE    ,DH101+10       ;TYPE PREV COMMAND HEADER
 9235  043164  104401  001315                  TYPE    ,$CRLF
 9236  043170  012701  000006                  MOV     #6,R1           ;SIX COMMAND VALUES
 9237  043174  012702  001236                  MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
 9238  043200  012246                  30$:     MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
 9239  043202  104402                          TYPOC                   ;TYPE IT
 9240  043204  104401  013252                  TYPE    ,SPACE2         ;TYPE SEPARATORS
 9241  043210  005301                          DEC     R1              ;SEE IF 7 VALUES TYPED YET
 9242  043212  001372                          BNE     30$             ;BR IF NOT
 9243  043214  104401  001315                  TYPE    ,$CRLF
 9244  043220  104401  013252                  TYPE    ,SPACE2         ;INDENT
 9245  043224  104401  063310                  TYPE    ,DH102          ;TYPE HDR FOR BA DATA
 9246  043230  104401  001315                  TYPE    ,$CRLF
 9247  043234  104401  013252                  TYPE    ,SPACE2         ;INDENT
 9248  043240  012246                          MOV     (R2)+,-(SP)
 9249  043242  104402                          TYPOC                   ;TYPE PREV. HI BA BITS
 9250  043244  104401  013252                  TYPE    ,SPACE2
 9251  043250  011246                          MOV     (R2),-(SP)
 9252  043252  104402                          TYPOC                   ;TYPE PREV. LO BA BITS
 9253  043254  104401  013236                  TYPE    ,CR2LF
```

```
 9254  043260  104401                        TYPE                    ;TYPE DATA HEADER
 9255  043262  000000                4$:     .WORD   0               ;DH POINTER GOES HERE
 9256  043264  104401  001315                TYPE    ,$CRLF
 9257  043270  005005                        CLR     R5              ;INIT INDENT INDICATOR
 9258  043272  032777  010000  135640  5$:   BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 9259  043300  001056                        BNE     20$             ;BR IF YES
 9260  043302  012001                        MOV     (R0)+,R1        ;GET DT POINTER
 9261  043304  001454                        BEQ     20$             ;BRANCH IF THERE ARE NONE
 9262  043306  012000                        MOV     (R0)+,R0        ;GET DF POINTER
 9263  043310  012002                        MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
 9264  043312  112003                10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
 9265  043314  105720                        TSTB    (R0)+           ;BUMP PAST FORMAT WORD
 9266  043316  005703                        TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
 9267  043320  001417                        BEQ     14$             ;NO - SKIP DATA PRINT
 9268  043322  005705                        TST     R5              ;SEE IF SHOULD INDENT
 9269  043324  001002                        BNE     11$             ;BR IF NOT
 9270  043326  104401  013252                TYPE    ,SPACE2         ;INDENT
 9271  043332  013146                11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
 9272  043334  104402                        TYPOC                   ;TYPE IT
 9273  043336  005303                        DEC     R3              ;MORE DATA WORDS
 9274  043340  001403                        BEQ     12$             ;NO-BRANCH
 9275  043342  104401  013252                TYPE    ,SPACE2         ;TYPE SEPARATORS
 9276  043346  000771                        BR      11$             ;LOOP
 9277  043350  005702                12$:    TST     R2              ;SEE IF <CR>,<LF> NEEDED
 9278  043352  001402                        BEQ     14$             ;BR IF NOT
 9279  043354  104401  001315                TYPE    ,$CRLF          ;TYPE IT
 9280  043360  005302                14$:    DEC     R2              ;MORE DH'S?
 9281  043362  003425                        BLE     20$             ;NO-BRANCH
 9282  043364  012037  043416        15$:    MOV     (R0)+,16$       ;GET NEXT DH POINTER
 9283  043370  105710                        TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
 9284  043372  001004                        BNE     34$             ;BR IF YES
 9285  043374  104401  001315                TYPE    ,$CRLF          ;SKIP EXTRA LINE
 9286  043400  005005                        CLR     R5              ;RE-INIT INDENT INDICATOR
 9287  043402  000404                        BR      36$
 9288  043404  005105                34$:    COM     R5              ;COMPLEMENT INDENT INDICATOR
 9289  043406  001002                        BNE     36$             ;BR IF NO INDENT REQUIRED
 9290  043410  104401  013252                TYPE    ,SPACE2         ;INDENT
 9291  043414  104401                36$:    TYPE                    ;TYPE DH
 9292  043416  000000                16$:    .WORD   0               ;DH POINTER GOES HERE
 9293  043420  104401  001315                TYPE    ,$CRLF
 9294  043424  105710                        TSTB    (R0)            ;TYPE A DT?
 9295  043426  001331                        BNE     10$             ;YES-BRANCH
 9296  043430  062700  000002                ADD     #2,R0           ;INCREMENT DF POINTER
 9297  043434  000751                        BR      14$             ;SEE IF END OF DF BLOCK
 9298  043436  104410                20$:    RESREG
 9299  043440  000207                        RTS     PC
 9300                          ;;*****************************************************************
 9301                          .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
 9302                          ;*ENTRY:        JSR PC, CONERR
 9303                          ;*RETURN:       RTS PC
 9304                          ;*
 9305                          ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
 9306                          ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
 9307                          ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
 9308                          ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
 9309                          ;*AT THIS TIME.
```

```
                                ;;*****************************************************************
043442  104407                  CONERR: SAVREG                          ;SAVE R0-R5
043444  152737  000377  003123          BISB    #377,DONE               ;SET DONE FLAG
043452  105237  003126                  INCB    ERRCNT                  ;INCREMENT ERROR COUNT
043456  004737  046344                  JSR     PC,TOPROC               ;LOAD RK REGS INTO $REGS
043462  032737  000001  003052          BIT     #BIT0,E.CONT            ;ERROR 0?
043470  001402                          BEQ     1$                      ;NO-BRANCH
043472  104064                          ERROR   64                      ;CLEAR CONT DID NOT CLEAR ERROR
043474  000474                          BR      7$
043476  032737  000002  003052  1$:     BIT     #BIT1,E.CONT            ;ERROR 1?
043504  001402                          BEQ     2$                      ;NO-BRANCH
043506  104065                          ERROR   65                      ;NO ATTENTION IN ATTENTION SUM REG
043510  000466                          BR      7$
043512  032737  000004  003052  2$:     BIT     #BIT2,E.CONT            ;ERROR 2?
043520  001407                          BEQ     3$                      ;NO-BRANCH
043522  105737  003136                  TSTB    DRNAFG                  ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
043526  001402                          BEQ     15$                     ;BR IF NOT
043530  105237  003137                  INCB    REISSU                  ;SET FLAG TO RE-ISSUE COMMAND
043534  104066                  15$:    ERROR   66                      ;UNSOLICITED ATTENTION
043536  000453                          BR      7$
043540  032737  000010  003052  3$:     BIT     #BIT3,E.CONT            ;ERROR 3?
043546  001402                          BEQ     4$                      ;NO-BRANCH
043550  104067                          ERROR   67                      ;UNEXPECTED DATA TYPE ERROR
043552  000445                          BR      7$
043554  032737  000020  003052  4$:     BIT     #BIT4,E.CONT            ;ERROR 4?
043562  001402                          BEQ     5$                      ;NO-BRANCH
043564  104070                          ERROR   70                      ;ATTENTION DID NOT RESET WITH CLEAR
043566  000437                          BR      7$
043570  032737  000040  003052  5$:     BIT     #BIT5,E.CONT            ;ERROR 5?
043576  001402                          BEQ     6$                      ;NO-BRANCH
043600  104071                          ERROR   71                      ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
043602  000431                          BR      7$
043604  032737  000400  003052  6$:     BIT     #BIT8,E.CONT
043612  001402                          BEQ     8$
043614  104072                          ERROR   72                      ;DATA LATE WHEN UNLOADING HEADER
043616  000423                          BR      7$
043620  032737  001000  003052  8$:     BIT     #BIT9,E.CONT
043626  001402                          BEQ     9$
043630  104073                          ERROR   73                      ;CONTROLLER ERROR DURING DRIVER SERVICE
043632  000415                          BR      7$
043634  032737  002000  003052  9$:     BIT     #BIT10,E.CONT
043642  001402                          BEQ     10$
043644  104074                          ERROR   74                      ;DRIVE DETECTED PARITY ERROR
043646  000407                          BR      7$
043650  032737  100000  003052  10$:    BIT     #BIT15,E.CONT
043656  001402                          BEQ     11$
043660  104052                          ERROR   52                      ;MULTIPLE DRIVE SELECT
043662  000401                          BR      7$
043664  104075                  11$:    ERROR   75                      ;UNDEFINED ERROR
043666  005037  003052          7$:     CLR     E.CONT                  ;CLEAR CONTROLLER ERROR WORD
043672  000137  046144                  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.
```

```
 9366  043676                          REPSUP:
 9367  043676  104407                          SAVREG
 9368  043700  005037  005476                  CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
 9369  043704  116537  000001  005476          MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
 9370  043712  012700  001162                  MOV     #$REG0,R0       ;FOR REPORTING
 9371  043716  016520  000002                  MOV     P.CYLN(R5),(R0)+
 9372  043722  116520  000005                  MOVB    P.TRCK(R5),(R0)+
 9373  043726  105020                          CLRB    (R0)+
 9374  043730  116520  000004                  MOVB    P.SECT(R5),(R0)+
 9375  043734  105020                          CLRB    (R0)+
 9376  043736  016520  000012                  MOV     P.WC(R5),(R0)+
 9377  043742  012700  001174                  MOV     #$REG5,R0
 9378  043746  116503  000007                  MOVB    P.BAHI(R5),R3
 9379  043752  042703  177774                  BIC     #177774,R3
 9380  043756  010337  001256                  MOV     R3,$REG36       ;HI BA BITS
 9381  043762  016537  000010  001260          MOV     P.BALO(R5),$REG37 ;LO BA BITS
 9382  043770  016520  000016                  MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
 9383  043774  016520  000020                  MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
 9384  044000  016520  000030                  MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
 9385  044004  016520  000026                  MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
 9386  044010  016520  000022                  MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
 9387                                                                  ;FOR ALL REPORTS (TO BE
 9388  044014  016520  000024                  MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
 9389  044020  016520  000032                  MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
 9390  044024  016520  000036                  MOV     P.DS(R5),(R0)+
 9391  044030  016520  000034                  MOV     P.ER(R5),(R0)+
 9392  044034  016520  000040                  MOV     P.A00(R5),(R0)+
 9393  044040  016520  000042                  MOV     P.B00(R5),(R0)+
 9394  044044  016520  000044                  MOV     P.A01(R5),(R0)+
 9395  044050  016520  000046                  MOV     P.B01(R5),(R0)+
 9396  044054  016520  000050                  MOV     P.A10(R5),(R0)+
 9397  044060  016520  000052                  MOV     P.B10(R5),(R0)+
 9398  044064  016520  000054                  MOV     P.A11(R5),(R0)+
 9399  044070  016520  000056                  MOV     P.B11(R5),(R0)+
 9400                              ;STORE PREVIOUS COMMAND FOR PRINTOUT
 9401  044074  012701  001236                  MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
 9402  044100  012700  005224                  MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
 9403  044104  112021                          MOVB    (R0)+,(R1)+     ;DRIVE NO.
 9404  044106  105021                          CLRB    (R1)+
 9405  044110  112021                          MOVB    (R0)+,(R1)+     ;COMMAND
 9406  044112  105021                          CLRB    (R1)+
 9407  044114  012021                          MOV     (R0)+,(R1)+     ;CYL ADDRESS
 9408  044116  116021  000001                  MOVB    1(R0),(R1)+     ;TRACK
 9409  044122  105021                          CLRB    (R1)+
 9410  044124  111021                          MOVB    (R0),(R1)+      ;SECTOR
 9411  044126  105021                          CLRB    (R1)+
 9412  044130  016021  000006                  MOV     6(R0),(R1)+     ;WORD COUNT
 9413  044134  116003  000003                  MOVB    3(R0),R3        ;HI BA BITS
 9414  044140  042703  177774                  BIC     #177774,R3
 9415  044144  010321                          MOV     R3,(R1)+
 9416  044146  016011  000004                  MOV     4(R0),(R1)      ;LO BA BITS
 9417  044152  104410                          RESREG
 9418  044154  000207                          RTS     PC
 9419                              ;;*****************************************************************
 9420                              .SBTTL  REPORT ERROR ROUTINE
 9421                              ;* ENTRY         JSR     PC, ERRHDL
```

```
                                        ;*RETURN        RTS     PC
                                        ;*
                                        ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
                                        ;*IN THE OPERATION.  THE ROUTINE DETERMINES WHICH COMMAND WAS
                                        ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
                                        ;;*****************************************************************

044156  104407                  ERRHDL: SAVREG
044160  152737  000377  003123          BISB    #377,DONE       ;SET DONE FLAG
044166  105237  003126                  INCB    ERRCNT          ;INCREMENT ERROR COUNT
044172  005037  005474                  CLR     RECODE          ;CLEAR RECOVERY CODE WORD
044176  032737  000400  005474  ER2ENT: BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
044204  001402                          BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
044206  012705  002714                  MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
044212  012737  046316  003046  52$:    MOV     #RETANL,A.ABNL  ;SET NOW ABNORMAL AND NORMAL RETURN FOR
044220  012737  046334  003044          MOV     #RETNML,A.NORM  ;DRIVER OPERATIONS IN ERROR PROCESSING
044226  004737  043676                  JSR     PC,REPSUP       ;GO SET UP REGISTERS FOR REPORT
                                                                ;NOW BEGIN TESTING THE ERROR
                                                                ;BITS. THE SEQUENCE IN
                                                                ;WHICH THEY ARE TESTED IS
                                                                ;CONSIDERED SIGNIFICANT IN
                                                                ;THAT ERRORS OF A MORE
                                                                ;CATASTROPHIC NATURE ARE FIRST
                                                                ;TESTED.
                                                                ;
                                                                ;IF AN ERROR IS FOUND SET,
                                                                ;THAT ERROR IS REPORTED AND
                                                                ;THE REPORTING IS TERMINATED.
                                                                ;IF ADDITIONAL ERRORS ARE SET,
                                                                ;THE RK611 REGISTER PRINTOUTS
                                                                ;WILL SHOW THIS BUT THE
                                                                ;REGISTER CONTENTS MUST BE
                                                                ;MANUALLY DECODED TO LOCATE THE
                                                                ;SECOND ERROR
044232  016504  000034                  MOV     P.ER(R5),R4     ;SET R4 TO ERROR REGISTER
044236  032765  020000  000020          BIT     #UPE,P.CS2(R5)  ;TEST UPE. IF UES-SET
044244  001406                          BEQ     1$              ;REPORT ERROR
044246  104001                          ERROR   1
044250  052737  000200  005474          BIS     #ABORT,RECODE
044256  000137  045652                  JMP     37$
044262  032765  004000  000020  1$:     BIT     #NEM,P.CS2(R5)  ;TEST NON-EXISTANT MEMORY
044270  001406                          BEQ     2$
044272  104002                          ERROR   2
044274  052737  000200  005474          BIS     #ABORT,RECODE
044302  000137  045652                  JMP     37$
044306  032765  010000  000020  2$:     BIT     #NED,P.CS2(R5)  ;TEST NON-EXISTANT DRIVE
044314  001412                          BEQ     3$
044316  032765  000400  000020          BIT     #UFE,P.CS2(R5)  ;TEST IF NED & UFE BOTH SET
044324  001403                          BEQ     38$
044326  104035                          ERROR   35              ;NED & UFE BOTH SET ERROR
044330  000137  045652                  JMP     37$
044334  104003                  38$:    ERROR   3               ;NED ONLY
044336  000137  045652                  JMP     37$
044342  032765  000400  000020  3$:     BIT     #UFE,P.CS2(R5)  ;TEST UNIT FIELD ERROR
044350  001412                          BEQ     4$
044352  032765  010000  000020          BIT     #NED,P.CS2(R5)  ;TEST IF UFE & NED BOTH SET
```

```
 9478  044360  001403                        BEQ     39$             ;NO-SKIP
 9479  044362  104035                        ERROR   35              ;REPORT NED & UFE BOTH SET
 9480  044364  000137  045652                JMP     37$
 9481  044370  104004              39$:      ERROR   4               ;REPORT UFE ONLY
 9482  044372  000137  045652                JMP     37$
 9483  044376  032765  000100  000014  4$:   BIT     #CMDTO,P.PRST(R5) ;
 9484  044404  001423                        BEQ     5$
 9485  044406  004737  046344                JSR     PC,TOPROC       ;GO PROCESS TIMEOUT
 9486  044412  032737  002000  005474        BIT     #TWOTOS,RECODE  ;2ND TIMEOUT IN TIMEOUT PROC?
 9487  044420  001007                        BNE     40$             ;YES - SKIP TO ERROR 56
 9488  044422  032737  000400  005474        BIT     #LEV2ER,RECODE  ;TEST IF LEVEL 2 ERROR
 9489  044430  001006                        BNE     41$             ;YES - SKIP TO ERROR 57
 9490  044432  104005                        ERROR   5               ;ELSE MAKE FULL TIMEOUT REPORT
 9491  044434  000137  045652                JMP     37$
 9492  044440  104056              40$:      ERROR   56              ;TWO TIMEOUTS ERROR REPORT
 9493  044442  000137  045652                JMP     37$
 9494  044446  104057              41$:      ERROR   57              ;2ND LEVEL ERROR REPORT
 9495  044450  000137  044176                JMP     ER2ENT          ;GO BUILD AND MAKE 2ND REPORT
 9496  044454  032765  010000  000014  5$:   BIT     #DRVSZD,P.PRST(R5) ;SEE IF DRIVE SIEZED BY OTHER PORT
 9497  044462  001403                        BEQ     65$             ;BR IF DRIVE IS AVAILABLE
 9498  044464  104107                        ERROR   107             ;DRIVE SIEZED BY OTHER PORT
 9499  044466  000137  045652                JMP     37$
 9500  044472  032765  020000  000016  65$:  BIT     #SPAR,P.CS1(R5) ;TEST D TO C PARITY ERROR
 9501  044500  001406                        BEQ     6$
 9502  044502  104006                        ERROR   6
 9503  044504  052737  004000  005474        BIS     #RCLREQ,RECODE
 9504  044512  000137  045652                JMP     37$
 9505  044516  032704  000010          6$:   BIT     #DRPAR,R4       ;TEST DRIVE DETECTED PARITY ERROR
 9506  044522  001406                        BEQ     7$
 9507  044524  104007                        ERROR   7
 9508  044526  052737  004000  005474        BIS     #RCLREQ,RECODE
 9509  044534  000137  045652                JMP     37$
 9510  044540  032765  000010  000036  7$:   BIT     #ACLO,P.DS(R5)  ;TEST AC LOW
 9511  044546  001403                        BEQ     8$
 9512  044550  104010                        ERROR   10
 9513  044552  000137  045652                JMP     37$
 9514  044556  032765  000020  000036  8$:   BIT     #SPDLSS,P.DS(R5) ;TEST SPEED LOSS
 9515  044564  001403                        BEQ     24$
 9516  044566  104011                        ERROR   11
 9517  044570  000137  045652                JMP     37$
 9518  044574  032765  004000  000016  24$:  BIT     #CTO,P.CS1(R5)  ;TEST FOR CONTROLLER TIMEOUT
 9519  044602  001401                        BEQ     25$
 9520  044604  104027                        ERROR   27
 9521  044606  032704  000001          25$:  BIT     #ILC,R4         ;TEST ILLEGAL FUNCTION CODE
 9522  044612  001403                        BEQ     10$
 9523  044614  104012                        ERROR   12
 9524  044616  000137  045652                JMP     37$
 9525  044622  032765  002000  000020  10$:  BIT     #PGE,P.CS2(R5)  ;TEST PROGRAMMING ERROR
 9526  044630  001403                        BEQ     11$
 9527  044632  104013                        ERROR   13
 9528  044634  000137  045652                JMP     37$
 9529  044640  032704  000004          11$:  BIT     #ILF,R4         ;TEST ILLEGAL DRIVE FUNCTION
 9530  044644  001403                        BEQ     12$
 9531  044646  104014                        ERROR   14
 9532  044650  000137  045652                JMP     37$
 9533  044654  032704  000040          12$:  BIT     #DTYE,R4        ;TEST DRIVE TYPE ERROR
```

```
 9534  044660  001403                        BEQ     13$
 9535  044662  104015                        ERROR   15
 9536  044664  000137  045652                JMP     37$
 9537  044670  032704  000020        13$:    BIT     #FMTE,R4        ;TEST FORMAT ERROR
 9538  044674  001403                        BEQ     14$
 9539  044676  104016                        ERROR   16
 9540  044700  000137  045652                JMP     37$
 9541  044704  032704  004000        14$:    BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
 9542  044710  001403                        BEQ     15$
 9543  044712  104017                        ERROR   17
 9544  044714  000137  045652                JMP     37$
 9545  044720  032704  040000        15$:    BIT     #UNS,R4         ;TEST DRIVE UNSAFE
 9546  044724  001406                        BEQ     16$
 9547  044726  104020                        ERROR   20
 9548  044730  052737  000200  005474        BIS     #ABORT,RECODE
 9549  044736  000137  045752                JMP     ALLTRM
 9550  044742  032704  000002        16$:    BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
 9551  044746  001406                        BEQ     17$
 9552  044750  104021                        ERROR   21
 9553  044752  052737  004000  005474        BIS     #RCLREQ,RECODE
 9554  044760  000137  045652                JMP     37$
 9555  044764  032704  001000        17$:    BIT     #COE,R4          TEST CYLINDER OVERFLOW
 9556  044770  001406                        BEQ     18$
 9557  044772  104022                        ERROR   22
 9558  044774  052737  004000  005474        BIS     #RCLREQ,RECODE
 9559  045002  000137  045652                JMP     37$
 9560  045006  032704  002000        18$:    BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
 9561  045012  001406                        BEQ     19$
 9562  045014  104023                        ERROR   23
 9563  045016  052737  004000  005474        BIS     #RCLREQ,RECODE
 9564  045024  000137  045652                JMP     37$
 9565  045030  032765  000040  000036  19$:   BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
 9566  045036  001406                        BEQ     20$
 9567  045040  104024                        ERROR   24
 9568  045042  052737  004000  005474        BIS     #RCLREQ,RECODE
 9569  045050  000137  045652                JMP     37$
 9570  045054  032704  010000        20$:    BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
 9571  045060  001406                        BEQ     21$
 9572  045062  104025                        ERROR   25
 9573  045064  052737  000200  005474        BIS     #ABORT,RECODE
 9574  045072  000137  045652                JMP     37$
 9575  045076  032765  100000  000020  21$:   BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
 9576  045104  001403                        BEQ     22$
 9577  045106  104026                        ERROR   26
 9578  045110  000137  045652                JMP     37$
 9579  045114  032704  020000        22$:    BIT     #OPI,R4         ;TEST IF OPI ERROR
 9580  045120  001457                        BEQ     29$
 9581  045122  052737  000010  005474        BIS     #OPIERR,RECODE
 9582  045130  105737  003121                TSTB    DERCNT          ;TEST IF FIRST ERROR
 9583  045134  001402                        BEQ     50$
 9584  045136  000137  045652                JMP     37$             ;NO - SKIP REPORT
 9585  045142  032737  000400  005474  50$:   BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
 9586  045150  001403                        BEQ     26$             ;HAS ALREADY OCCURRED
 9587  045152  104054                27$:    ERROR   54
 9588  045154  000137  045652                JMP     37$             ;GET OUT OF ERROR REPORT
 9589  045160  004737  047004        26$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
```

```
 9590  045164  004737  046514                 JSR     PC,RDHD0          ;GET HEADER OF SECTOR 0
 9591  045170  032737  000400  005474         BIT     #LEV2ER,RECODE    ;TEST IF ERROR GETTING HDR
 9592  045176  001025                         BNE     28$               ;IF YES-GO MAKE ABREVIATED REPORT
 9593  045200  013702  003036                 MOV     RKBAS,R2          ;STORE HEADER 0 INTO REGISTERS
 9594  045204  042762  000100  000000         BIC     #IE,RKCS1(R2)     ;RESET INTERRUPT ENABLE
 9595  045212  016237  000024  001202         MOV     RKDB(R2),$REG10   ;FOR REPORTING
 9596  045220  016237  000024  001204         MOV     RKDB(R2),$REG11
 9597  045226  016237  000024  001206         MOV     RKDB(R2),$REG12
 9598  045234  032762  100000  000000         BIT     #CERR,RKCS1(R2)   ;TEST IF ERROR DURING STORAGE
 9599  045242  001343                         BNE     27$
 9600  045244  104030                         ERROR   30                ;MAKE OPI REPORT
 9601  045246  000137  045652                 JMP     37$
 9602  045252  104054             28$:        ERROR   54
 9603  045254  000137  044176                 JMP     ER2ENT            ;GO MAKE 2ND LEVEL REPORT
 9604  045260  032704  000400             29$:BIT     #HVRC,R4          ;TEST IF HVRC ERROR
 9605  045264  001457                         BEQ     23$
 9606  045266  052737  000004  005474         BIS     #HVRCER,RECODE
 9607  045274  105737  003121                 TSTB    DERCNT            ;TEST IF FIRST ERROR
 9608  045300  001402                         BEQ     30$               ;YES - REPORT
 9609  045302  000137  045652                 JMP     37$               ;JUMP TO RETURN
 9610  045306  032737  000400  005474 30$:    BIT     #LEV2ER,RECODE    ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
 9611  045314  001403                         BEQ     31$               ;OCCURRED. NO-SKIP EXIT
 9612  045316  104055             51$:        ERROR   55
 9613  045320  000137  045652                 JMP     37$               ;GET OUT OF ERROR REPORT
 9614  045324  004737  047004             31$:JSR     PC,BLDEXH         ;GO BUILD EXPECTED HEADER
 9615  045330  004737  046514                 JSR     PC,RDHD0          ;GO GET HDR 0
 9616  045334  032737  000400  005474         BIT     #LEV2ER,RECODE    ;TEST IF ERROR IN GETTING HDR
 9617  045342  001025                         BNE     32$               ;IF YES-GO MAKE ABREVIATED REPORT
 9618  045344  013702  003036                 MOV     RKBAS,R2          ;GET RK611 BASE ADDRESS
 9619  045350  042762  000100  000000         BIC     #IE,RKCS1(R2)     ;CLEAR INTERRUPT ENABLE
 9620  045356  016237  000024  001202         MOV     RKDB(R2),$REG10   ;STORE OFF HEADER
 9621  045364  016237  000024  001204         MOV     RKDB(R2),$REG11
 9622  045372  016237  000024  001206         MOV     RKDB(R2),$REG12
 9623  045400  032762  100000  000000         BIT     #CERR,RKCS1(R2)   ;TEST IN ANY ERROR IN UNLOAD
 9624  045406  001343                         BNE     51$               ;IY YES - GO MAKE SHORT REPORT
 9625  045410  104031                         ERROR   31                ;MAKE FULL REPORT
 9626  045412  000137  045652                 JMP     37$
 9627  045416  104055             32$:        ERROR   55                ;ABREVIATED HVRC ERROR RPORT
 9628  045420  000137  044176                 JMP     ER2ENT            ;GO REPORT 2ND LEVEL ERROR
 9629  045424  032704  000200             23$:BIT     #BSE,R4           ;TEST FOR BAD SECTOR ERROR
 9630  045430  001430                         BEQ     33$               ;NO - SKIP
 9631  045432  052737  000002  005474         BIS     #BSERR,RECODE     ;SET ERROR FLAG
 9632  045440  016500  000030                 MOV     P.DCYL(R5),R0     ;GET CYL NO.
 9633  045444  116501  000027                 MOVB    P.DTS+1(R5),R1    ;GET TRACK NO.
 9634  045450  042701  177774                 BIC     #177774,R1        ;CLEAR ALL BITS EXCEPT TRACK
 9635  045454  016502  000026                 MOV     P.DTS(R5),R2      ;GET SECTOR IN ERROR
 9636  045460  042702  177740                 BIC     #177740,R2        ;CLEAR ALL BITS EXCEPT SECTOR
 9637  045464  004737  042404                 JSR     PC,BDSRCK         ;GO SEE IF THIS SECTOR LISTED
 9638  045470  032737  001000  005474         BIT     #BADSEC,RECODE    ;TEST RESULT
 9639  045476  001065                         BNE     37$               ;YES - EXIT, NO ERROR OR REPORT
 9640  045500  104104                         ERROR   104               ;ELSE REPORT BAD BSE
 9641  045502  042737  000002  005474         BIC     #BSERR,RECODE     ;RESET BSE ERROR FLAG
 9642  045510  000460                         BR      37$               ;GO EXIT
 9643  045512  032704  100000             33$:BIT     #DCK,R4           ;TEST IF DATA CHECK
 9644  045516  001435                         BEQ     36$
 9645  045520  052737  000020  005474         BIS     #DCKERR,RECODE    ;SET DATA CHECK ERROR IN RECOVERY CODE
```

```
 9646  045526  032704  000100                    BIT     #ECH,R4          ;TEST IF ECC IS HARD. IF
 9647  045532  001406                            BEQ     34$              ;YES SET UNCORRECTABLE IN
 9648  045534  052737  000040  005474            BIS     #ECCNC,RECODE    ;RECOVERY FLAG AND A 0 IN
 9649  045542  005037  001206                    CLR     $REG12           ;REG12 TO INDICATE UNCORRECTABLE,
 9650  045546  000403                            BR      35$              ;A 1 IN REG 12 FOR CORRECTABLE
 9651  045550  012737  000001  001206  34$:     MOV     #1,$REG12
 9652  045556  105737  003121          35$:     TSTB    DERCNT           ;TEST IF FIRST ERROR
 9653  045562  001033                            BNE     37$              ;NO SKIP REPORT
 9654  045564  004737  046660                    JSR     PC,GTPKAD        ;GO GET PACK ADDRESS OF ERROR
 9655  045570  016537  000060  001202            MOV     P.EPOS(R5),$REG10 ;STORE ECC POSITION &
 9656  045576  016537  000062  001204            MOV     P.EPAT(R5),$REG11 ;PATTERN
 9657  045604  104032                            ERROR   32               ;REPORT DCK ERROR
 9658  045606  000137  045652                    JMP     37$
 9659  045612  032765  040000  000020  36$:     BIT     #WCE,P.CS2(R5)   ;TEST WRITE CHECK ERROR
 9660  045620  001414                            BEQ     37$
 9661  045622  042737  000200  005474            BIC     #ABORT,RECODE    ;CLEAR ABORT & SET WRITE
 9662  045630  052737  000100  005474            BIS     #WCERR,RECODE    ;CHECK ERROR IN RECODE
 9663  045636  105737  003121                    TSTB    DERCNT           ;TEST IF FIRST ERROR
 9664  045642  001003                            BNE     37$              ;NO - SKIP
 9665  045644  004737  046660                    JSR     PC,GTPKAD        ;GO GET ADDRESS OF ERROR
 9666  045650  104033                            ERROR   33               ;REPORT WCE
 9667
 9668  045652  032765  000020  000014  37$:     BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
 9669  045660  001404                            BEQ     43$
 9670  045662  104036                            ERROR   36
 9671  045664  052737  000200  005474            BIS     #ABORT,RECODE
 9672
 9673  045672  032765  000040  000014  43$:     BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
 9674  045700  001404                            BEQ     44$
 9675  045702  104037                            ERROR   37
 9676  045704  052737  000200  005474            BIS     #ABORT,RECODE
 9677
 9678  045712  032765  004000  000014  44$:     BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
 9679  045720  001404                            BEQ     46$
 9680  045722  104040                            ERROR   40
 9681  045724  052737  000200  005474            BIS     #ABORT,RECODE
 9682
 9683  045732  032765  000010  000014  46$:     BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
 9684  045740  001404                            BEQ     ALLTRM
 9685  045742  104042                            ERROR   42
 9686  045744  052737  000200  005474            BIS     #ABORT,RECODE
 9687
 9688
 9689                                    ;ALL ERRORS MUST EXIT THROUGH THIS POINT
 9690
 9691  045752  012705  002630          ALLTRM:  MOV     #PARM0,R5        ;RESTORE PARAMETER BLOCK SELECTION
 9692  045756  012737  044156  003046            MOV     #ERRHDL,A.ABNL   ;RESTORE INTERRUPT VECTORS FOR RETRY
 9693  045764  012737  042700  003044            MOV     #ERRFFE,A.NORM   ;DRIVER OPERATIONS, IF ANY
 9694  045772  032737  000200  005474            BIT     #ABORT,RECODE    ;IF ABORT IS NOT SET AND
 9695  046000  001043                            BNE     48$              ;THE DRIVE IS READY (HAS NOT
 9696                                                                     ;CYCLED DOWN)
 9697  046002  013702  003036                    MOV     RKBAS,R2         ;GET BASE ADDRESS
 9698  046006  032762  000200  000012            BIT     #RDY,RKDS(R2)    ;TEST IF DRIVE READY SET
 9699  046014  001004                            BNE     47$              ;RECALIBRATE REQUIRED BIT IS SET
 9700  046016  052737  000200  005474            BIS     #ABORT,RECODE    ;ELSE ABORT WITH ABORT MESSAGE
 9701  046024  000431                            BR      48$
```

```
 9702  046026  032737  004000  005474  47$:    BIT     #RCLREQ,RECODE  ;IF RECALIBRATE IS REQUIRED
 9703  046034  001443                          BEQ     BGNRTY          ;FOR RETRY SET UP PARAM
 9704  046036  112737  000113  000001          MOVB    #RECAL,P.CMND   ;BLOCK TO DO IT.
 9705  046044  012737  046316  003046          MOV     #RETANL,A.ABNL
 9706  046052  004737  042510                  JSR     PC,DRVCAL
 9707  046056  012737  044156  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR RETURN
 9708  046064  032737  000400  005474          BIT     #LEV2ER,RECODE  ;IF AN ERROR OCCURRED IN THE
 9709  046072  001424                          BEQ     BGNRTY          ;RECAL ATTEMP SET ABORT,
 9710  046074  052737  000200  005474          BIS     #ABORT,RECODE   ;PRINT THE RECAL ERROR MESSAGE, AND
 9711  046102  104060                          ERROR   60              ;GO REPORT DETAILS
 9712  046104  000137  044176                  JMP     ER2ENT
 9713  046110  104061                  49$:    ERROR   61              ;REPORT ABORT-RETRY FAILED
 9714  046112  105037  003130                  CLRB    DRVERS          ;CLR DRV ERR CNT
 9715  046116  105037  003126                  CLRB    ERRCNT
 9716  046122  000137  017272                  JMP     NEWDRV          ;TEST NEXT DRV
 9717
 9718                                  ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
 9719                                  ;IF THE DRIVE READY BIT IS RESET.
 9720
 9721  046126  000005                  HLTPRG: RESET                   ;DISABLE ALL DEVICES
 9722  046130  000000                          HALT                    ;HALT THE CPU
 9723  046132  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 9724  046136  000005                          RESET                   ;RESET ALL DEVICES
 9725  046140  000137  013350                  JMP     CMSTRT
 9726
 9727
 9728                                  ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
 9729                                  ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
 9730                                  ; THE INITIATING ROUTINE FOR RETRY.  ANY OTHER ERROR IS TO BE
 9731                                  ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
 9732                                  ;FLAG IS SET AND PROGRAM HALTS.
 9733
 9734  046144  032737  000136  005474  BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
 9735  046152  001404                          BEQ     3$
 9736  046154  052737  100000  005474  9$:     BIS     #ANYDER,RECODE
 9737  046162  000453                          BR      2$
 9738  046164                          3$:
 9739  046164  105737  003142                  TSTB    NORTRY          ;SEE IF "NO-RETRY" FLAG SET
 9740  046170  001371                          BNE     9$              ;BR IF YES
 9741  046172  032737  001000  005474          BIT     #BADSEC,RECODE  ;TEST IF BAD SECTOR FLAG SET
 9742  046200  001044                          BNE     2$              ;IF YES-EXIT TO CALLER
 9743  046202  123737  003126  003127          CMPB    ERRCNT,ERRLMT   ;BEGIN RETRY IF ERROR COUNT HAS
 9744  046210  001012                          BNE     1$              ;NOT BEEN EXCEEDED
 9745  046212  005037  001174                  CLR     $REG5
 9746  046216  113737  003126  001174          MOVB    ERRCNT,$REG5    ;GET ERROR RETRY COUNT
 9747  046224  104102                          ERROR   102             ;REPORT RETRY UNSUCCESSFUL
 9748  046226  052737  000200  005474          BIS     #ABORT,RECODE   ;SET ABORT & QUIT
 9749  046234  000734                          BR      HLTPRG
 9750  046236  013702  003036          1$:     MOV     RKBAS,R2        ;GET RK BASE ADDRESS
 9751  046242  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
 9752  046250  004737  042510                  JSR     PC,DRVCAL       ;GO DO IT
 9753  046254  012700  005240                  MOV     #COMSTR,R0      ;GO AND REESTABLISH THE COMMAND
 9754  046260  012025                          MOV     (R0)+,(R5)+     ;AS IT WAS ENTERED INTO THE
 9755  046262  012025                          MOV     (R0)+,(R5)+     ;PARAMETER BLOCK
 9756  046264  012025                          MOV     (R0)+,(R5)+
 9757  046266  012025                          MOV     (R0)+,(R5)+
```

```
 9758  046270  012025                          MOV     (R0)+,(R5)+
 9759  046272  012025                          MOV     (R0)+,(R5)+
 9760  046274  012705  002630                  MOV     #PARM0,R5
 9761  046300  012737  000000  177776          MOV     #PR0,@#PSW      ;LOWER PRIORITY TO ALLOW INTERRUPT
 9762  046306  004737  042510                  JSR     PC,DRVCAL       ;CALL DRIVER
 9763  046312  104410                  2$:     RESREG                  ;IF RETURN GETS HERE, NO ERROR
 9764  046314  000207                          RTS     PC              ;OCCURRED, RECOVERY WAS SUCCESSFUL
 9765
 9766
 9767  046316  152737  000377  003123  RETANL: BISB    #377,DONE       ;SET DONE
 9768  046324  052737  000400  005474          BIS     #LEV2ER,RECODE  ;SET LEVEL TWO ERROR
 9769  046332  000207                          RTS     PC
 9770  046334  152737  000377  003123  RETNML: BISB    #377,DONE       ;SET DONE
 9771  046342  000207                          RTS     PC
 9772
 9773
 9774                                  ;;*****************************************************************
 9775                                  .SBTTL  TIME OUT PROCESSOR ROUTINE
 9776                                  ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
 9777                                  ;*GATHERING DUTIES.
 9778                                  ;;*****************************************************************
 9779  046344                          TOPROC:
 9780  046344  104407                          SAVREG
 9781  046346  013702  003036                  MOV     RKBAS,R2
 9782  046352  012701  001174                  MOV     #$REG5,R1       ;SET UP R1 FOR RK REGISTER STORAGE
 9783  046356  016221  000000                  MOV     RKCS1(R2),(P1)+ ;STORE ALL VALID RK611
 9784  046362  016221  000010                  MOV     RKCS2(R2),(R1)+ ;REGISTERS
 9785  046366  016221  000020                  MOV     RKDC(R2),(R1)+
 9786  046372  016221  000006                  MOV     RKDA(R2),(R1)+
 9787  046376  016221  000002                  MOV     RKWC(R2),(R1)+
 9788  046402  016221  000004                  MOV     RKBA(R2),(R1)+
 9789  046406  016221  000016                  MOV     RKASOF(R2),(R1)+
 9790  046412  016221  000012                  MOV     RKDS(R2),(R1)+
 9791  046416  016221  000014                  MOV     RKER(R2),(R1)+
 9792  046422  005000                          CLR     R0              ;THIS CODE WILL ATTEMPT TO
 9793                                                                  ;RETRIEVE THE STATUS FROM THE
 9794                                                                  ;DRIVE.
 9795  046424  012705  002714                  MOV     #PARM1,R5       ;SET UP TO USE PARM1
 9796  046430  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
 9797  046436  004737  042510                  JSR     PC,DRVCAL       ;CALL DRIVER
 9798  046442  032765  000100  000014          BIT     #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
 9799  046450  001403                          BEQ     1$              ;NO - SKIP
 9800  046452  052737  002000  005474          BIS     #TWOTOS,RECODE  ;SET TWO TIMEOUTS FLAG
 9801  046460  062705  000040          1$:     ADD     #P.A00,R5       ;BUMP R5 TO POINT TO DRIVE STATUS
 9802  046464  012521                          MOV     (R5)+,(R1)+     ;MOVE ALL THE DRIVE STATUS INTO THE
 9803  046466  012521                          MOV     (R5)+,(R1)+     ;TEMP REGS FOR REPORTING.
 9804  046470  012521                          MOV     (R5)+,(R1)+
 9805  046472  012521                          MOV     (R5)+,(R1)+
 9806  046474  012521                          MOV     (R5)+,(R1)+
 9807  046476  012521                          MOV     (R5)+,(R1)+
 9808  046500  012521                          MOV     (R5)+,(R1)+
 9809  046502  012521                          MOV     (R5)+,(R1)+
 9810
 9811  046504  012705  002630                  MOV     #PARM0,R5       ;RESTORE PARM 0
 9812  046510  104410                          RESREG
 9813  046512  000207                          RTS     PC
```

```
                        ;;*****************************************************************
                        .SBTTL  READ HEADER 0 ROUTINE
                        ;;*****************************************************************
046514                  RDHD0:
046514  104407                  SAVREG
046516  016501  000026          MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
046522  016500  000052          MOV     P.B10(R5),R0    ;GET THE CYLINDER ADRS
046526  042700  160017          BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
046532  006200                  ASR     R0              ;OFF UNUSED BITS AND POSITION
046534  006200                  ASR     R0              ;FOR USE AS THE DESIRED
046536  006200                  ASR     R0              ;CYLINDER IN THE READ
046540  006200                  ASR     R0              ;HEADER COMMAND.
046542  012705  002714          MOV     #PARM1,R5       ;SET UP TO USE P.B. 1
046546  010165  000004          MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
046552  010065  000002          MOV     R0,P.CYLN(R5)   ;SET CYL NO.
046556  112765  000177  000001  MOVB    #SUBCLR,P.CMND(R5) ;SET S.S. CLEAR CMND
046564  012737  000000  177776  MOV     #PR0,@#PSW      ;ALLOW INTERRUPTS
046572  004737  042510          JSR     PC,DRVCAL       ;CLEAR THE S.S.
046576  112765  000125  000001  MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
046604  004737  042510          JSR     PC,DRVCAL       ;DO READ HEADER
046610  013712  020004          MOV     HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT = 0
046614  032712  000200  4$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
046620  001775                  BEQ     4$              ;BR IF NOT RDY YET
046622  013712  020006          MOV     HOLD4,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
046626  032712  000200  6$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
046632  001775                  BEQ     6$              ;BR IF NOT RDY YET
046634  142765  000020  000007  BICB    #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
046642  153765  003125  000007  BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
046650  012705  002630          MOV     #PARM0,R5       ;RESTORE P.B. 0 ADDRESS
046654  104410                  RESREG                  ;RESTORE R0-R5
046656  000207                  RTS     PC              ;RETURN


                        ;;*****************************************************************
                        .SBTTL  GET PACK ADDRESS ROUTINE
                        ;;*****************************************************************
046660  016537  000030  001174  GTPKAD: MOV     P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
046666  005037  001176          CLR     $REG6           ;CLEAR REGISTERS FOR
046672  005037  001200          CLR     $REG7           ;TRACK & SECTOR STORAGE
046676  116537  000026  001200  MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
046704  116537  000027  001176  MOVB    P.DTS+1(R5),$REG6
046712  005737  001200          TST     $REG7            ;ADJUST THE ADDRESS CONTAINED IN
                                                        ;THE RK REGISTERS FOR THE AUTOMATIC
046716  001403                  BEQ     1$              ;INCREMENT
046720  005337  001200          DEC     $REG7
046724  000426                  BR      5$
046726  032765  010000  000016  1$:     BIT     #CFMT,P.CS1(R5)
046734  001404                  BEQ     2$
046736  012737  000023  001200          MOV     #19.,$REG7
046744  000403                  BR      3$
046746  012737  000025  001200  2$:     MOV     #21.,$REG7
046754  005737  001176  3$:     TST     $REG6
```

```
 9870  046760  001403                          BEQ     4$
 9871  046762  005337  001176                  DEC     $REG6
 9872  046766  000405                          BR      5$
 9873  046770  012737  000002  001176  4$:     MOV     #2,$REG6
 9874  046776  005337  001174                  DEC     $REG5
 9875  047002  000207                  5$:     RTS     PC
 9876
 9877
 9878
 9879                                  ;;*****************************************************************
 9880                                  .SBTTL  BUILD EXPECTED HEADER
 9881                                  ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
 9882                                  ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
 9883                                  ;*7 FOR REPORTING.
 9884                                  ;;*****************************************************************
 9885  047004  104407                  BLDEXH: SAVREG
 9886  047006  016537  000030  001174          MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
 9887  047014  016501  000026                  MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
 9888  047020  042701  174377                  BIC     #174377,R1      ;CLEAR ALL BUT TRACK BITS
 9889  047024  006201                          ASR     R1              ;AND SECTOR. SHIFT THE TRACK
 9890  047026  006201                          ASR     R1              ;OVER TO CONFORM TO HEADER FORMAT
 9891  047030  006201                          ASR     R1              ;CHECK THE FORMAT BIT AND
 9892                                                                  ;IF SET, SET THE HEADER FORMAT
 9893  047032  016537  000026  001176          MOV     P.DTS(R5),$REG6 ;BIT.
 9894  047040  042737  177740  001176          BIC     #177740,$REG6   ;CLEAR ALL BUT SECTOR
 9895  047046  060137  001176                  ADD     R1,$REG6        ;ADD TRACK AND SECTOR TOGETHER
 9896  047052  052737  140000  001176          BIS     #140000,$REG6   ;INSERT BSE BITS
 9897  047060  032765  010000  000016          BIT     #CFMT,P.CS1(R5)
 9898  047066  001403                          BEQ     23$
 9899  047070  052737  001000  001176          BIS     #1000,$REG6
 9900  047076  013737  001174  001200  23$:     MOV     $REG5,$REG7     ;COMPUTE THE HEADER VRC
 9901  047104  013701  001176                  MOV     $REG6,R1
 9902  047110  043737  001176  001200          BIC     $REG6,$REG7
 9903  047116  043701  001174                  BIC     $REG5,R1
 9904  047122  050137  001200                  BIS     R1,$REG7
 9905  047126  104410                          RESREG
 9906  047130  000207                          RTS     PC
 9907
```

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

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

                                .SBTTL   *WATCH-DOG TIMER

                                ;.*****************************************************************
                                ;*
                                ;*      THE WATCH-DOG TIMER DOES A PSEUDO-TIMING OF RK06-RK07 UNIBUS
                                ;*      SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
                                ;*      REAL-TIME CLOCK (KW11-P OR KW11-L) IS ON THE SYSTEM
                                ;*      THE RK06-RK07 DRIVER WILL USE THE LOCATION W.MTIM FOR
                                ;*      MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
                                ;*      WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
                                ;*      DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
                                ;*      (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
                                ;*      IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
                                ;*      TIME-OUT WILL BE DESIGNATED IN THE PROGRAM DEVICE STATUS
                                ;*      REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
                                ;*
                                ;*      THE DRIVER WILL USE THE LOCATION W.MIN AS THE NUMBER
                                ;*      OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
                                ;*      THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
                                ;*      LIMIT FOR ALL OTHER COMMANDS.
                                ;*
                                ;*      FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
                                ;*      WATCH UP TO 8 OPERATIONS SIMULATEOUSLY. FOR SEQUENTIAL
                                ;*      OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
                                ;*
                                ;*CALL  JSR     PC,W.WTCH
                                ;*      RETURN          IF NO DRIVE ORDER EXCEEDED ITS TIME LIMIT
                                ;*
                                ;*      OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
                                ;*      BY LOCATION A.ABNL WILL OCCUR AND THE CMDTO FLAG
                                ;*      IN THE PROGRAM DEVICE STATUS REGISTER OF THE
                                ;*      APPROPRIATE PARAMETER BLOCK WILL BE SET.
                                ;*
                                ;;*****************************************************************

 047132  010546          W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
 047134  010446                  MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
 047136  010346                  MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
 047140  010246                  MOV     R2,-(SP)        ;SAVE R2 ON STACK
 047142  013746  177776          MOV     P.,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
 047146  005337  003056          DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
 047152  001034                  BNE     20$             ;IF NOT ZERO RETURN
 047154  013737  003060  003056  MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
 047162  105737  003100          TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
 047166  001426                  BEQ     20$             ;NO, RETURN
 047170  013737  003042  177776  MOV     RKPRI,PS        ;LOCK OUT RK06-RK07 INTERRUPTS
 047176  013702  003036          MOV     RKBAS,R2        ;LOAD BASE OF RK06-RK07 REGISTERS
 047202  005337  003114          DEC     W.DRV           ;DECREMENT COMMAND TIMER
 047206  001016                  BNE     20$             ;RETURN IF NO TIME OUT
```

```
 9964  047210  105037  003100                  CLRB    W.TIME          ;RESET TIMING INDICATOR
 9965  047214  013705  003112                  MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
 9966                                                                  ;  TABLE FOR INDEXING
 9967  047220  052765  000100  000014          BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
 9968  047226  020537  003054                  CMP     R5,O.WAIT       ;CHECK IF DRIVER IS WAITING FOR
 9969                                                                  ;  COMMAND COMPLETION
 9970  047232  001002                          BNE     5$              ;NO, DO NOT ALTER WAITING FOR
 9971                                                                  ;  COMMAND COMPLETION
 9972  047234  005037  003054                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9973  047240  004737  052514          5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
 9974  047244  012637  177776          20$:    MOV     (SP)+,PS        ;RESTORE PSW
 9975  047250  012602                          MOV     (SP)+,R2        ;RESTORE R2
 9976  047252  012603                          MOV     (SP)+,R3        ;RESTORE R3
 9977  047254  012604                          MOV     (SP)+,R4        ;RESTORE R4
 9978  047256  012605                          MOV     (SP)+,R5        ;RESTORE R5
 9979  047260  000207                          RTS     PC              ;RETURN
```

```
                        .SBTTL   *RK06 INTERRUPT SERVICE ROUTINE

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

047262  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
047264  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
047266  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
047270  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
047272  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
047274  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
047276  013702  003036          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
047302  016237  000010  003002  MOV     RKCS2(R2),T.CS2 ;STORE CS2
047310  032737  001000  003002  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
047316  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
047320  052737  100000  003052  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
047326  004737  052540          JSR     PC,R.CONT       ;REPORT ERROR
```

```
10036  047332  000137  051512                    JMP     I.RTRN          ;RETURN
10037
10038  047336  105737  003074          1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
10039  047342  001410                             BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
10040  047344  100403                             BMI     5$              ;CHECK IF RELEASE COMMAND
10041  047346  105037  003074                     CLRB    I.ISRL          ;YES,CLEAR FLAG
10042  047352  000473                             BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
10043
10044  047354  105037  003074          5$:       CLRB    I.ISRL          ;CLEAR FLAG
10045  047360  000137  050474                     JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
10046
10047  047364  032737  010400  003002  6$:       BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
10048                                                                     ;  UNIT FIELD ERROR
10049  047372  001413                             BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
10050  047374  013704  003002                     MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
10051  047400  042704  177770                     BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
10052  047404  013705  003112                     MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
10053  047410  016237  000000  003000             MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
10054  047416  000137  047704                     JMP     I.ERRC          ;REPORT ERROR
10055
10056  047422  016237  000012  003020  7$:       MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
10057  047430  032737  000001  003020             BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
10058                                                                     ;  PORT
10059  047436  001041                             BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
10060
10061                                             ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
10062  047440  032737  164000  003002             BIT     #DLT!WCE!UPE!NEM,T.CS2
10063
10064  047446  001007                             BNE     10$             ;INDICATE ERROR
10065  047450  016237  000014  003016             MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
10066
10067                                   ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
10068  047456  032737  125700  003016             BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10069
10070  047464  001407                             BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
10071
10072  047466  052737  000010  003052  10$:       BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
10073  047474  004737  052540                     JSR     PC,R.CONT       ;REPORT ERROR
10074  047500  000137  051512                     JMP     I.RTRN          ;RESTORE REGISTERS
10075
10076  047504  105037  003100          11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
10077  047510  005037  003114                     CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
10078  047514  013705  003112                     MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
10079                                                                     ;  ADDRESS
10080  047520  052765  010000  000014             BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
10081                                                                     ;  PROGRAM DRIVE STATUS REGISTER
10082  047526  005037  003054                     CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
10083  047532  004737  052514                     JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
10084  047536  000137  051512                     JMP     I.RTRN          ;GO RESTORE REGISTERS
10085
10086  047542  013705  003054          I.IOO:    MOV     O.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
10087  047546  001002                             BNE     2$              ;IS COMMAND WAITING PROCESSING
10088                                                                     ;  YES, DO PROCESSING
10089  047550  000137  050474                     JMP     I.ATTN          ;NO, PROCESS ATTENTION
10090
10091  047554  013704  003002          2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
10092  047560  042704  177770                    BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
10093
10094
10095  047564  126504  000000                    CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
10096  047570  001401                            BEQ     3$              ;YES, CONTINUE
10097  047572  000000                            HALT                    ;NO, DRIVER ERROR
10098  047574  122765  000164  000001  3$:       CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
10099  047602  001002                            BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
10100  047604  000137  050142                    JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
10101
10102  047610  005037  003054          10$:      CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
10103  047614  005037  003114                    CLR     W.DRV           ;CLEAR WATCH-DOG TIME
10104  047620  105037  003100                    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
10105  047624  016237  000000  003000            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
10106  047632  032737  100000  003000            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
10107  047640  001021                            BNE     I.ERRC          ;YES, PROCESS ERROR
10108  047642  016237  000016  003014            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10109  047650  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
10110  047656  001004                            BNE     15$             ;YES, REPORT ERROR
10111  047660  004737  052526                    JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
10112  047664  000137  051512                    JMP     I.RTRN          ;RESTORE REGISTERS
10113
10114  047670  052765  000010  000014  15$:      BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
10115
10116  047676  004737  052162          I.ERRA:   JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
10117  047702  000405                            BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
10118
10119  047704  013765  003000  000016  I.ERRC:   MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
10120  047712  004737  052204                    JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
10121  047716  016265  000032  000062  I.ERR:    MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
10122  047724  016265  000030  000060            MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
10123  047732  004037  051530                    JSR     R0,I.CCLR       ;CLEAR CONTROLLER
10124  047736  051512                            I.RTRN                  ;ERROR RETURN
10125  047740  032765  010400  000020            BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
10126                                                                    ;  UNIT FIELD ERROR
10127  047746  001046                            BNE     5$              ;YES, REPORT ERROR
10128  047750  004037  052266                    JSR     R0,I.STAT       ;GATHER DRIVE STATUS
10129  047754  051512                            I.RTRN                  ;ERROR RETURN
10130  047756  112737  000005  003000            MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
10131  047764  004037  051612                    JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
10132  047770  051512                            I.RTRN                  ;ERROR RETURN
10133  047772  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
10134  050000  001407                            BEQ     2$              ;NO, INDICATE DRIVE ERROR
10135  050002  052737  000020  003052            BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
10136                                                                    ;  WITH CLEAR
10137  050010  004737  052540                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
10138  050014  000137  051512                    JMP     I.RTRN          ;GO RESTORE REGISTERS
10139
10140  050020  032737  040000  003024  2$:       BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
10141  050026  001403                            BEQ     3$              ;YES, CHECK FAULT
10142  050030  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
10143  050036  032737  001000  003026  3$:       BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
10144  050044  001407                            BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
10145  050046  052737  002000  003052            BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
10146  050054  004737  052540                    JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
10147  050060  000137  051512                    JMP     I.RTRN          ;RETURN
```

```
10148
10149  050064  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
10150  050072  001017                          BNE     10$                ;YES, GO REPORT ERROR
10151  050074  032737  020000  003024          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS CYCLING DOWN
10152  050102  001413                          BEQ     10$                ;NO, REPORT ERROR
10153  050104  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
10154  050112  113737  003101  003100          MOVB    INTMSK,W.TIME      ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
10155  050120  013737  003064  003114          MOV     W.8SEC,W.DRV
10156  050126  000137  051512                  JMP     I.RTRN             ;GO RESTORE REGISTERS
10157
10158  050132  004737  052514          10$:    JSR     PC,R.ABNL          ;GO REPORT ERROR
10159  050136  000137  051512                  JMP     I.RTRN             ;GO RESTORE REGISTERS
10160
10161                                  .SBTTL   *READ ALL HEADERS INTERRUPT SEQUENCE
10162
10163  050142  016237  000000  003000  I.HDAL. MOV     RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLEP
10164                                                                  ;  ERROR
10165  050150  032737  100000  003000          BIT     #CERR,T.CS1        ;CHECK IF CONTROLLER ERROR
10166  050156  001422                          BEQ     5$                 ;NO, CHECK FOR ATTENTION
10167
10168  050160  005037  003054                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETE
10169  050164  105037  003100                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
10170  050170  005037  003114                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
10171  050174  013765  003000  000016          MOV     T.CS1,P.CS1(R5)    ;STORE ERROR RKCS1
10172  050202  004737  052204                  JSR     PC,I.CST1          ;STORE CONTROLLER REGISTERS
10173  050206  004037  051530                  JSR     R0,I.CCLR          ;CLEAR CONTROLLER
10174  050212  051512                          I.RTRN                     ;ERROR RETURN
10175  050214  004737  052514                  JSR     PC,R.ABNL          ;INDICATE ERROR RETURN
10176  050220  000137  051512                  JMP     I.RTRN             ;RESTORE REGISTERS
10177
10178  050224  016237  000016  003014  5$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10179  050232  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
10180  050240  001410                          BEQ     7$                 ;NO, CHECK IF READ ALL HEADERS
10181  050242  005037  003054                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETION
10182  050246  105037  003100                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
10183  050252  005037  003114                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
10184  050256  000137  047676                  JMP     I.ERRA             ;GO REPORT ERROR
10185
10186  050262  013701  003070          7$:     MOV     HDR.AD,R1          ;GET MAIN MEMORY ADDRESS
10187  050266  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET FIRST WORD OF HEADER
10188  050272  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET SECOND WORD OF HEADER
10189  050276  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET THIRD WORD OF HEADER
10190  050302  010137  003070                  MOV     R1,HDR.AD          ;STORE ADDRESS FOR NEXT HEADER
10191  050306  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
10192  050314  032737  100000  003002          BIT     #DLT,T.CS2         ;CHECK FOR DATA LATE
10193  050322  001055                          BNE     35$                ;YES, REPORT ERROR
10194  050324  005337  003072                  DEC     HDR.CT             ;DECREMENT NUMBER OF HEADER YET TO READ
10195  050330  001026                          BNE     25$                ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
10196  050332  005037  003054                  CLR     O.WAIT             ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
10197  050336  005037  003114                  CLR     W.DRV              ;CLEAR TIME OUT COUNT FOR THIS DRIVE
10198  050342  105037  003100                  CLRB    W.TIME             ;CLEAR WATCH DOG TIME ON THIS DRIVE
10199  050346  012762  000003  000026          MOV     #3,RKMR1(R2)       ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
10200  050354  112737  000001  003000          MOVB    #DR.SEL,T.CS1      ;LOAD SELECT COMMAND
10201  050362  004037  051612                  JSR     R0,I.ISSU          ;GET SECTOR COUNT
10202  050366  051512                          I.RTRN                     ;ERROR RETURN
10203  050370  013765  003026  000056          MOV     T.MR3,P.B11(R5)    ;LOAD SECTOR COUNT
```

```
10204  050376  004737  052526                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
10205  050402  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10206
10207  050406  016562  000002  000020  25$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
10208  050414  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10209  050422  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10210  050430  042765  165777  000016          BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
10211                                                                  ;  DRIVE TYPE
10212  050436  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
10213  050444  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
10214  050452  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10215
10216  050456  052737  000400  003052  35$:    BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
10217  050464  004737  052540                  JSR     PC,R.CONT       ;REPORT ERROR
10218  050470  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10219
10220                                  .SBTTL   *DRIVE ATTENTION SCANNER
10221
10222  050474  016237  000000  003000  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
10223                                                                  ;  REGISTER 1 FOR COMPARISON
10224  050502  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
10225  050510  001441                          BEQ     5$              ;NO, CHECK IF ATTENTION
10226
10227                                          ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
10228  050512  032737  164000  003002          BIT     #DLT!WCE!UPE!NEM,T.CS2
10229
10230  050520  001007                          BNE     1$              ;INDICATE ERROR
10231  050522  016237  000014  003016          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
10232
10233                                  ;       CHECK FOR DATA TRANSFER ERROR TYPE
10234  050530  032737  125700  003016          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10235
10236  050536  001407                          BEQ     2$              ;NO DATA TRANSFER ERROR
10237
10238  050540  052737  000010  003052  1$:     BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
10239  050546  004737  052540                  JSR     PC,R.CONT       ;REPORT ERROR
10240  050552  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10241
10242  050556  013704  003002          2$:     MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
10243  050562  042704  177770                  BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
10244  050566  105037  003100                  CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
10245  050572  005037  003114                  CLR     W.DRV           ;RESET TIMER VALUE
10246  050576  013705  003112                  MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
10247
10248                                  ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
10249                                  ;         IN PROGRAM DEVICE STATUS REGISTER
10250  050602  042765  000006  000014          BIC     #DRVPOS!DRVPDT,P.PRST(R5)
10251
10252  050610  000137  047704                  JMP     I.ERRC          ;GO REPORT ERROR
10253
10254  050614  032737  040000  003000  5$:     BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
10255  050622  001002                          BNE     6$              ;YES, PROCESS INTERRUPT
10256  050624  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10257
10258  050630  016237  000016  003014  6$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10259  050636  105737  003015                  TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
10260  050642  001007                          BNE     7$              ;YES GO PROCESS INTERRUPT
10261  050644  052737  000002  003052          BIS     #E.NOAT,E.CONT  ;SET NO ATTENTION IN ATTENTION SUMMARY
10262  050652  004737  052540                  JSR     PC,R.CONT       ;GO REPORT ERROR
10263  050656  000137  051512                  JMP     I.RTRN          ;GO RESTORE REGISTERS
10264
10265  050662  133737  003101  003015  7$:     BITB    INTMSK,T.ASOF+1 ;CHECK IF DESIRED INTERRUPT
10266  050670  001007                          BNE     8$              ;YES, GO PROCESS IT
10267  050672  052737  000004  003052          BIS     #E.UATT,E.CONT  ;SET UNSOLICATED ATTENTION
10268  050700  004737  052540                  JSR     PC,R.CONT       ;GO REPORT ERROR
10269  050704  000137  051512                  JMP     I.RTRN          ;GO RESTORE REGISTERS
10270
10271  050710  013705  003112          8$:     MOV     PBLKT,R5        ;STORE PARAMETER BLOCK TABLE
10272  050714  116504  000000                  MOVB    P.DRVN(R5),R4   ;STORE DRIVE NUMBER
10273  050720  032765  020000  000014          BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
10274  050726  001402                          BEQ     11$             ;NO, CONTINUE
10275  050730  000137  051412                  JMP     I.UNLD          ;SERVICE DRIVE IN POSITION AFTER ERROR
10276
10277  050734  042765  000002  000014  11$:     BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
10278  050742  005062  000026                  CLR     RKMR1(R2)       ;CLEAR MAINTENANCE REGISTER 1
10279  050746  112737  000001  003000          MOVB    #DR.SEL,T.CS1   ;LOAD COMMAND
10280  050754  004037  051612                  JSR     R0,I.ISSU       ;SELECT DRIVE WITH ATTENTION HIGH
10281  050760  051512                          I.RTRN                  ;ERROR RETURN
10282  050762  013765  003026  000042          MOV     T.MR3,P.B00(R5) ;STORE STATUS BYTE 00 MESS B
10283  050770  032765  000200  000042          BIT     #S.FLT,P.B00(R5) ;CHECK IF DRIVE FAULT
10284  050776  001401                          BEQ     12$             ;NO, CHECK FOR DRIVE STATUS CHANGE
10285  051000  000461                          BR      I.AERR          ;PROCESS ERROR
10286
10287  051002  013765  003024  000040  12$:     MOV     T.MR2,P.A00(R5) ;STORE MAINTENANCE REGISTER 2
10288  051010  032765  040000  000040          BIT     #S.DSC,P.A00(R5) ;CHECK FOR DRIVE STATUS CHANGE
10289  051016  001004                          BNE     13$             ;YES, PROCESS DRIVE STATUS CHANGE
10290  051020  052765  004000  000014          BIS     #NODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
10291  051026  000446                          BR      I.AERR          ;PROCESS ERROR
10292
10293  051030  112737  000005  003000  13$:     MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
10294  051036  004037  051612                  JSR     R0,I.ISSU       ;CLEAR DRIVE STATUS CHANGE
10295  051042  051512                          I.RTRN                  ;ERROR RETUN
10296  051044  013765  003014  000032          MOV     T.ASOF,P.ASOF(R5) ;STORE ATTENTION SUMMARY
10297  051052  133765  003101  000033          BITB    INTMSK,P.ASOF+1(R5) ;CHECK IF ATTENTION RESET
10298  051060  001407                          BEQ     15$             ;YES, CONTINUE INTERRUPT PROCESSING
10299  051062  052737  000020  003052          BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
10300                                                                  ;  WITH DRIVE CLEAR
10301  051070  004737  052540                  JSR     PC,R.CONT       ;FLAG ERROR
10302  051074  000137  051512                  JMP     I.RTRN          ;RESTORE REGISTERS
10303
10304  051100  013765  003024  000040  15$:     MOV     T.MR2,P.A00(R5) ;STORE MAINTENANCE REGISTER 2
10305  051106  032765  040000  000040          BIT     #S.DSC,P.A00(R5) ;CHECK IF DRIVE STATUS CHANGE
10306                                                                  ;  RESET
10307  051114  001404                          BEQ     16$             ;YES, CONTINUE INTERRUPT PROCESSING
10308  051116  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
10309  051124  000407                          BR      I.AERR          ;GO PROCESS ERROR
10310
10311  051126  105037  003100          16$:     CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
10312  051132  005037  003114                  CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
10313  051136  004737  052526                  JSR     PC,R.NORM       ;REPORT SUCCESSFUL COMMAND COMPLETETION
10314  051142  000563                          BR      I.RTRN          ;RESTORE REGISTERS
10315
```

```
10316                                   .SBTTL   *ATTENTION ERROR HANDLER
10317
10318  051144  042765  000004  000014  I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
10319                                                                    ;  OF DATA TRANSFER
10320  051152  105037  003100                  CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
10321  051156  005037  003114                  CLR     W.DRV              ;RESET WATCH-DOG TIME
10322  051162  042765  177741  000016          BIC     #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
10323  051170  042737  000036  003000          BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
10324  051176  053765  003000  000016          BIS     T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
10325  051204  013765  003002  000020          MOV     T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
10326  051212  013765  003004  000022          MOV     T.WCR,P.WCR(R5)
10327  051220  013765  003006  000024          MOV     T.BA,P.BAR(R5)
10328  051226  013765  003010  000026          MOV     T.DA,P.DTS(R5)
10329  051234  013765  003012  000030          MOV     T.DC,P.DCYL(R5)
10330  051242  013765  003014  000032          MOV     T.ASOF,P.ASOF(R5)
10331  051250  013765  003016  000034          MOV     T.ER,P.ER(R5)
10332  051256  013765  003020  000036          MOV     T.DS,P.DS(R5)
10333  051264  004037  052266                  JSR     R0,I.STAT          ;GATHER DRIVE STATUS
10334  051270  051512                          I.RTRN                     ;ERROR RETURN
10335  051272  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD COMMAND
10336  051300  004037  051612                  JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
10337  051304  051512                          I.RTRN                     ;ERROR RETURN
10338  051306  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
10339  051314  001407                          BEQ     2$                 ;YES, FLAG DRIVE ERROR
10340  051316  052737  000020  003052          BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
10341  051324  004737  052540                  JSR     PC,R.CONT          ;REPORT ERROR
10342  051330  000137  051512                  JMP     I.RTRN             ;RESTORE REGISTERS
10343
10344  051334  032765  000020  000014  2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
10345  051342  001017                          BNE     10$                ;YES, REPORT ERROR
10346  051344  032737  020000  003024          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
10347  051352  001413                          BEQ     10$                ;NO, REPORT ERROR
10348  051354  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
10349  051362  113737  003101  003100          MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
10350  051370  013737  003064  003114          MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
10351  051376  000137  051512                  JMP     I.RTRN             ;RESTORE REGISTERS
10352
10353  051402  004737  052514          10$:    JSR     PC,R.ABNL          ;REPORT ERROR
10354  051406  000137  051512                  JMP     I.RTRN             ;RESTORE REGISTERS
10355
10356                                   .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD
10357
10358  051412  052765  020000  000014  I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
10359  051420  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
10360  051426  004037  051612                  JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
10361  051432  051512                          I.RTRN                     ;ERROR RETURN
10362  051434  136437  003101  003015          BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
10363  051442  001406                          BEQ     15$                ;YES, CONTINUE
10364  051444  012737  000020  003052          MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
10365  051452  004737  052540                  JSR     PC,R.CONT          ;REPORT ERROR
10366  051456  000415                          BR      I.RTRN             ;RESTORE REGISTERS
10367
10368  051460  032737  040000  003024  15$:     BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
10369  051466  001403                          BEQ     20$                ;YES, CONTINUE
10370  051470  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
10371  051476  105037  003100          20$:     CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

```
10372  051502  005037  003114                  CLR     W.DRV           ;CLEAR TIME COUNT
10373  051506  004737  052514                  JSR     PC,R.ABNL       ;REPORT ERROR
10374
10375  051512  012600                  I.RTRN: MOV     (SP)+,R0        ;RESTORE R0
10376  051514  012601                          MOV     (SP)+,R1        ;RESTORE R1
10377  051516  012602                          MOV     (SP)+,R2        ;RESTORE R2
10378  051520  012603                          MOV     (SP)+,R3        ;RESTORE R3
10379  051522  012604                          MOV     (SP)+,R4        ;RESTORE R4
10380  051524  012605                          MOV     (SP)+,R5        ;RESTORE R5
10381  051526  000002                          RTI                     ;RETURN
10382
```

```
                                .SBTTL   *CONTROLLER CLEAR ROUTINE

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

051530  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
051536  016237  000000  003000          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
051544  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
051552  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
051554  052737  000001  003052          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
051562  004737  052540                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
051566  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
051570  000200                          RTS     R0              ;RETURN

051572  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
051600  112737  177777  003074          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
051606  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
051610  000200                          RTS     R0              ;RETURN
```

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

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

051612  013746  003000          I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
051616  005037  003002                  CLR     T.CS2           ;CLEAR TEMPORARY CS2
051622  116537  000000  003002          MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
051630  013762  003002  000010          MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
051636  116537  000007  003001          MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
051644  142737  177753  003001          BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                                                ;  FORMAT AND DRIVE TYPE
051652  013762  003000  000000          MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
051660  105762  000000          1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
051664  100375                          BPL     1$
051666  004737  052034                  JSR     PC,I.STOR       ;GO STORE REGISTERS
051672  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
051700  001437                          BEQ     5$              ;NO, RETURN
051702  032737  001000  003002          BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
051710  001406                          BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
051712  052737  100000  003052          BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
051720  004737  052540                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
051724  000440                          BR      10$             ;RETURN

                                        ;CHECK IF ANY CONTROLLER ERROR IS SET
051726  032737  024000  003000  2$:     BIT     #CTO!SPAR,T.CS1
051734  001027                          BNE     7$
051736  032737  176400  003002          BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
051744  001023                          BNE     7$
051746  032737  131761  003016          BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
051754  001017                          BNE     7$

051756  122716  000005                  CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
10474  051762  001003                        BNE     3$              ;NO, DO NOT SET DRIVE HARD ERROR
10475  051764  052765  000020  000014        BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
10476  051772  004037  051530          3$:   JSR     R0,I.CCLR       ;GO ISSUE A CONTROLLER CLEAR
10477  051776  052026                        10$                     ;ERROR RETURN
10478  052000  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10479  052006  005726                        TST     (SP)+           ;ADJUST STACK
10480  052010  005720                        TST     (R0)+           ;ADJUST R0 FOR NORMAL RETURN
10481  052012  000200                        RTS     R0              ;RETURN
10482
10483  052014  052737  001000  003012  7$:   BIS     #E.CERR,E.CONT  ;SET CONTROLLER ERROR DURING
10484                                                                ;  DRIVER SERVICING
10485  052022  004737  052540                JSR     PC,R.CONT       ;REPORT ERROR
10486  052026  005726                  10$:  TST     (SP)+           ;ADJUST STACK
10487  052030  011000                        MOV     (R0),R0         ;ADJUST R0 FOR ERROR RETURN
10488  052032  000200                        RTS     R0              ;RETURN
```

```
10489                                   .SBTTL   *STORE RK611 UNIBUS REGISTERS
10490
10491                                   ;;*****************************************************************
10492                                   ;*
10493                                   ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL
10494                                   ;*      RK611 REGISTER IN TEMPORARY LOCATIONS.
10495                                   ;*
10496                                   ;*CALL  JSR     PC,I.STOR
10497                                   ;*      RETURN
10498                                   ;*
10499                                   ;*      REGISTER        USE
10500                                   ;*      --------        ---
10501                                   ;*
10502                                   ;*      R2              ADDRESS OF RK611 REGISTERS
10503                                   ;*
10504                                   ;*****************************************************************
10505
10506  052034  016237  000000  003000  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
10507  052042  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
10508  052050  016237  000002  003004          MOV     RKWC(R2),T.WCR
10509  052056  016237  000004  003006          MOV     RKBA(R2),T.BA
10510  052064  016237  000006  003010          MOV     RKDA(R2),T.DA
10511  052072  016237  000012  003020          MOV     RKDS(R2),T.DS
10512  052100  016237  000014  003016          MOV     RKER(R2),T.ER
10513  052106  016237  000016  003014          MOV     RKASOF(R2),T.ASOF
10514  052114  016237  000020  003012          MOV     RKDCYL(R2),T.DC
10515  052122  016237  000026  003022          MOV     RKMR1(R2),T.MR1
10516  052130  016237  000034  003024          MOV     RKMR2(R2),T.MR2
10517  052136  016237  000036  003026          MOV     RKMR3(R2),T.MR3
10518  052144  016237  000030  003030          MOV     RKECPS(R2),T.POS
10519  052152  016237  000032  003032          MOV     RKECPT(R2),T.PAT
10520  052160  000207                          RTS     PC              ;RETURN
```

```
10521                                   .SBTTL   *STORE CONTROLLER STATUS
10522
10523                                   ;;*****************************************************************
10524                                   ;*
10525                                   ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
10526                                   ;*      THE FOLLOWING REGISTERS WILL BE STORED:
10527                                   ;*
10528                                   ;*              COMMAND AND STATUS REGISTER 2
10529                                   ;*              WORD COUNT REGISTER
10530                                   ;*              BUS ADDRESS REGISTER
10531                                   ;*              DESIRED TRACK AND SECTOR
10532                                   ;*              STATUS REGISTER
10533                                   ;*              ERROR REGISTER
10534                                   ;*              ATTENTION SUMMARY/OFFSET REGISTER
10535                                   ;*              CYLINDER ADDRESS REGISTER
10536                                   ;*
10537                                   ;*CALL  JSR     PC,I.CSTS
10538                                   ;*      RETURN
10539                                   ;*
10540                                   ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
10541                                   ;*
10542                                   ;*
10543                                   ;*              REGISTER                CONTENTS
10544                                   ;*              --------                --------
10545                                   ;*
10546                                   ;*              R2                      RK06 BASE ADDRESS
10547                                   ;*              R5                      ADDRESS OF PARAMETER BLOCK
10548                                   ;*
10549                                   ;;*****************************************************************
10550
10551  052162  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
10552                                                                          ;OF LAST COMMAND ISSUED
10553  052170  042737  000036  003000          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
10554  052176  053765  003000  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
10555  052204  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
10556  052212  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
10557  052220  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
10558  052226  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
10559  052234  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
10560  052242  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
10561  052250  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
10562                                                                          ;  OFFSET
10563  052256  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
10564  052264  000207                          RTS     PC                      ;RETURN
10565
```

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

052266  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 01
052274  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
052302  004037  051612                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
052306  052476                          3$                              ;ERROR RETURN
052310  013765  003024  000044          MOV     T.MR2,P.A0(R5)          ;STORE STATUS BYTE 01 MESS A
052316  013765  003026  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
052324  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 10
052332  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
052340  004037  051612                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
052344  052476                          3$                              ;ERROR RETURN
052346  013765  003024  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
052354  013765  003026  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
052362  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
                                                                        ;  FOR STATUS BYTE 11
052370  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
052376  004037  051612                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
052402  052476                          3$                              ;ERROR RETURN
052404  013765  003024  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
052412  013765  003026  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
052420  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
                                                                        ;  FOR STATUS BYTE 00
052424  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
052432  004037  051612                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
052436  052476                          3$                              ;ERROR RETURN
052440  013765  003024  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
052446  013765  003026  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
052454  032737  001000  003026          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
052462  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
10622  052464  052737  002000  003052          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
10623  052472  004737  052540                  JSR     PC,R.CONT               ;REPORT ERROR
10624  052476  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
10625  052500  000200                          RTS     R0                      ;RETURN
10626
10627  052502  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
10628  052510  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
10629  052512  000200                          RTS     R0                      ;RETURN
10630
```

```
10631                                   .SBTTL   *COMMON DRIVER RETUR
10632
10633  052514  105037  003101          R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10634  052520  004777  130322                  JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
10635  052524  000207                          RTS     PC              ;RETURN
10636
10637  052526  105037  003101          R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10638  052532  004777  130306                  JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
10639  052536  000207                          RTS     PC              ;RETURN
10640
10641  052540  105037  003101          R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
10642  052544  105037  003100                  CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
10643  052550  005037  003114                  CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
10644  052554  004777  130270                  JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
10645  052560  000207                          RTS     PC              ;RETURN
```

```
                        .SBTTL   *COMMAND INITATOR

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

052562  010546          C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
052564  010446                  MOV     R4,-(SP)        ;STORE R4 ON STACK
052566  010346                  MOV     R3,-(SP)        ;STORE R3 ON STACK
052570  010246                  MOV     R2,-(SP)        ;STORE R2 ON STACK
052572  010146                  MOV     R1,-(SP)        ;STORE R1 ON STACK
052574  010046                  MOV     R0,-(SP)        ;STORE R0 ON STACK
052576  013746  177776          MOV     PS,-(SP)        ;STORE PSW ON STACK
052602  013737  003042  177776  MOV     RKPRI,PS        ;LOCK OUT RK0c INTERRUPTS
052610  017605  000016          MOV     @16(SP),R5      ;STORE PARAME ER BLOCK ADDRESS
052614  062766  000002  000016  ADD     #2,16(SP)       ;ADJUST RETURN
052622  016504  000000          MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
052626  042704  177770          BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
052632  010537  003112          MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
052636  116437  003102  003101  MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
052644  116437  003102  003100  MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
052652  013737  003062  003114  MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

052660  013702  003036          MOV     RKBAS,R2        ;LOAD R2 WITH RK06 ADDRESS BASE

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

```
10702  052664  042765  075176  000014          BIC     #^C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
10703
10704  052672  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
10705  052674  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
10706  052700  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
10707  052702  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
10708
10709  052706  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
10710  052710  020001                          CMP     R0,R1           ;CHECK IF FINISHED
10711  052712  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
10712  052714  105037  003074                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
10713  052720  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
10714  052724  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
10715  052730  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
10716  052736  001402                          BEQ     3$              ;NO, PROCESS
10717  052740  000137  053454                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
10718
10719  052744  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
10720                                                                  ;  START SPINDLE
10721                                                                  ;  RECALIBRATE
10722                                                                  ;  OFFSET
10723                                                                  ;  SEEK
10724                                                                  ;  UNLOAD
10725
10726  052752  101174                          BHI     25$             ;NO, DRIVE COMMAND
10727                                                                  ;  SELECT DRIVE
10728                                                                  ;  PACK ACKNOWLEDGE
10729                                                                  ;  CLEAR
10730
10731  052754  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
10732  052762  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
10733                                                                  ;  READ DATA
10734                                                                  ;  WRITE DATA
10735                                                                  ;  READ HEADER
10736                                                                  ;  WRITE HEADER
10737                                                                  ;  WRITE CHECK
10738  052764  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10739  052772  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
10740  053000  005037  003054                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND
10741  053004  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
10742  053012  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
10743  053014  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10744  053022  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10745  053030  000431                          BR      8$              ;GO ISSUE COMMAND
10746
10747  053032  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
10748  053040  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
10749  053042  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
10750  053050  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
10751  053056  000416                          BR      8$              ;GO ISSUE COMMAND
10752
10753  053060  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
10754  053066  001003                          BNE     7$              ;NO, CHECK IF RECAL
10755  053070  013737  003066  003114          MOV     W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
10756  053076  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
10757  053104  001003                          BNE     8$              ;NO, CONTINUE
```

```
10758  053106  013737  003064  003114          MOV     W.8SEC,W.DRV    ;LOAD RECAL TIME FOR 8 SECONDS
10759  053114  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10760  053122  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10761                                                                  ;  AND DRIVE TYPE
10762  053130  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10763  053136  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10764  053144  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10765  053152  001533                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
10766  053154  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
10767  053162  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10768  053170  004737  047132          10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
10769  053174  016237  000000  003000          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
10770  053202  032737  000200  003000          BIT     #RDY,T.CS1      ;WAIT FOR READY
10771  053210  001767                          BEQ     10$
10772  053212  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK FOR ERROR
10773  053220  001011                          BNE     15$             ;YES, GIVE NORMAL RETURN
10774  053222  004737  047132          11$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
10775  053226  016237  000016  003014          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10776  053234  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
10777  053242  001767                          BEQ     11$             ;WAIT FOR DRIVE INTERRUPT
10778  053244  105037  003100          15$:    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
10779  053250  005037  003114                  CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
10780  053254  004737  052526                  JSR     PC,R.NORM       ;INDICATE COMMAND IS FINISHED
10781  053260  000137  054434                  JMP     C.RTRN          ;RESTORE REGISTERS
10782
10783  053264  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
10784  053272  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
10785  053300  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
10786  053306  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
10787  053314  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5)  ;CHECK IF WRITE CHECK COMMAND
10788  053322  001010                          BNE     25$             ;NO, GO ISSUE THE COMMAND
10789  053324  032765  000200  000014          BIT     #W.WCK,P.PRST(R5)  ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
10790  053332  001404                          BEQ     25$             ;NO, GO ISSUE THE COMMAND
10791  053334  012765  000123  000016          MOV     #WRDATA,P.CS1(R5)  ;ISSUE WRITE COMMAND
10792  053342  000406                          BR      26$             ;GO ISSUE COMMAND
10793
10794  053344  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10795  053352  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10796  053360  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10797  053366  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
10798                                                                  ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
10799                                                                  ;  BITS 16-17
10800  053374  010537  003054                  MOV     R5,O.WAIT       ;LOAD WAITING FOR COMMAND
10801  053400  032765  100000  000014          BIT     #DTBA11,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
10802  053406  001403                          BEQ     27$             ;NO, LOAD CS2
10803  053410  052765  000020  000020          BIS     #BA1,P.CS2(R5)  ;SET INHIBIT BUS ADDRESS INCREMENT
10804  053416  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
10805  053424  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10806  053432  001403                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
10807  053434  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
10808  053442  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10809  053450  000137  054434                  JMP     C.RTRN          ;RESTORE REGISTERS
10810
10811                                  .SBTTL   *SPECIAL COMMAND PROCESSING
10812
10813  053454  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
10814  053462  001132                        BNE     10$             ;NO, PROCESS OTHER COMMANDS
10815  053464  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
10816  053472  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10817  053500  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10818                                                                ;  AND DRIVE TYPE
10819  053506  112765  000001  000016        MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
10820  053514  016562  000016  000000        MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10821  053522  004737  047132          2$:   JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
10822  053526  016265  000000  000016        MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
10823  053534  032765  000200  000016        BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
10824  053542  001767                        BEQ     2$
10825  053544  004737  052204                JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
10826  053550  016265  000034  000040        MOV     RKMR2(R2),P.A00(R5) ;STORE STATUS BYTE 00 MESSAGE A
10827  053556  016265  000036  000042        MOV     RKMR3(R2),P.B00(R5) ;STORE STATUS BYTE 00 MESSAGE B
10828  053564  032765  100000  000016        BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
10829  053572  001436                        BEQ     6$              ;NO, GATHER DRIVE STATUS
10830  053574  105037  003100                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
10831  053600  005037  003114                CLR     W.DRV           ;CLEAR WATCH DOG COUNT
10832  053604  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10833  053612  001043                        BNE     8$              ;YES, INDICATE NORMAL RETURN
10834  053614  032765  001000  000020        BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
10835  053622  001043                        BNE     9$              ;YES, INDICATE CONTROLLER ERROR
10836  053624  004037  051530                JSR     R0,I.CCLR       ;CLEAR ERROR
10837  053630  054434                        C.RTRN                  ;ERROR RETURN
10838  053632  032765  010400  000020        BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
10839  053640  001007                        BNE     5$              ;REPORT ERROR
10840  053642  032765  000001  000036        BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
10841  053650  001003                        BNE     5$              ;YES, REPORT ERROR
10842  053652  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
10843  053660  004737  052514          5$:   JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
10844  053664  000137  054434                JMP     C.RTRN          ;RESTORE REGISTERS
10845
10846  053670  004037  052266          6$:   JSR     R0,I.STAT       ;GATHER DRIVE STATUS
10847  053674  054434                        C.RTRN                  ;ERROR RETURN
10848  053676  105037  003100                CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
10849  053702  005037  003114                CLR     W.DRV           ;RESET WATCH-DOG TIME
10850  053706  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10851  053714  001402                        BEQ     8$              ;NO, REPORT ERROR
10852  053716  005062  000000                CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
10853  053722  004737  052526          8$:   JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
10854  053726  000137  054434                JMP     C.RTRN          ;RESTORE REGISTERS
10855
10856  053732  052737  100000  003052  9$:   BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
10857  053740  004737  052540                JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
10858  053744  000137  054434                JMP     C.RTRN
10859
10860  053750  122765  000140  000001  10$:  CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
10861  053756  001040                        BNE     13$             ;NO, CHECK IF READ ALL HEADERS
10862  053760  010537  003054                MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
10863                                                                ;  WAIT FOR COMMAND
10864  053764  052765  000010  000020        BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
10865  053772  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
10866  054000  112737  000001  003074        MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
10867  054006  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
10868  054014  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10869                                                                ;  AND DRIVE TYPE
```

```
10870  054022  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
10871  054030  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10872  054036  001403                          BEQ     11$               ;NO, DO NOT RESET INTERRUPT ENABLE
10873  054040  042765  000100  000016          BIC     #IE,P.CS1(R5)     ;RESET INTERRUPT ENABLE
10874  054046  016562  000016  000000  11$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10875  054054  000137  054434                  JMP     C.RTRN            ;RESTORE REGISTERS
10876
10877  054060  122765  000164  000001  13$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
10878  054066  001053                          BNE     30$               ;NO, CHECK IF CONTROLLER CLEAR
10879  054070  010537  003054                  MOV     R5,O.WAIT         ;SET WAITING FOR COMMAND COMMPLETION
10880  054074  016537  000010  003070          MOV     P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
10881  054102  132765  000020  000007          BITB    #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
10882  054110  001404                          BEQ     14$               ;YES, LOAD 22 IN HEADER COUNT
10883  054112  012737  000024  003072          MOV     #20.,HDR.CT       ;LOAD 20 IN SECTOR COUNT
10884  054120  000403                          BR      22$               ;GO ISSUE READ HEADER COMMAND
10885
10886  054122  012737  000026  003072  14$:    MOV     #22.,HDR.CT       ;LOAD 22 IN SECTOR COUNT
10887  054130  016562  000002  000020  22$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10888  054136  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
10889  054144  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10890  054152  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10891  054160  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
10892                                                                  ;  AND FORMAT
10893  054166  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
10894  054174  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10895  054202  001027                          BNE     34$               ;YES, INDICATE ILLEGAL DRIVER COMMAND
10896  054204  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
10897  054212  000137  054434                  JMP     C.RTRN            ;RESTORE REGISTERS
10898
10899  054216  122765  000176  000001  30$:    CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
10900  054224  001012                          BNE     32$               ;NO, CHECK IF SUBSYSTEM CLEAR
10901  054226  004037  051530                  JSR     R0,I.CCLR         ;CLEAR CONTROLLER
10902  054232  054434                          C.RTRN                    ;ERROR RETURN
10903  054234  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10904  054242  001472                          BEQ     40$               ;NO, INDICATE NORMAL RETURN
10905  054244  005062  000000                  CLR     RKCS1(R2)         ;RESET INTERRUPT ENABLE
10906  054250  000467                          BR      40$               ;INDICATE NORMAL RETURN
10907
10908  054252  122765  000177  000001  32$:    CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
10909  054260  001406                          BEQ     36$               ;YES, CLEAR SUBSYSTEM
10910  054262  052737  000100  003052  34$:    BIS     #E.ILLD,E.CONT    ;SET ILLEGAL DRIVER COMMAND
10911  054270  004737  052540                  JSR     PC,R.CONT         ;REPORT ERROR
10912  054274  000457                          BR      C.RTRN            ;RESTORE REGISTERS
10913
10914  054276  012762  000040  000010  36$:    MOV     #SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
10915  054304  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
10916  054312  032765  100000  000016          BIT     #CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
10917  054320  001406                          BEQ     37$               ;NO, FINISH COMMAND
10918  054322  052737  000001  003052          BIS     #BIT0,E.CONT      ;SET CLEAR SUBSYSTEM DID NOT CLEAR
10919                                                                  ;  CONTROLLER ERROR
10920  054330  004737  052540                  JSR     PC,R.CONT         ;REPORT ERROR
10921  054334  000437                          BR      C.RTRN            ;RESTORE REGISTERS
10922
10923  054336  013746  003060          37$:    MOV     W.MILI,-(SP)      ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
10924                                                                  ;  TO DISAPPEAR
10925  054342  016265  000000  000016  38$:    MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
10926  054350  032765  040000  000016          BIT     #DI,P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
10927  054356  001411                          BEQ     39$             ;YES, FINISH COMMAND
10928  054360  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
10929  054362  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
10930  054364  005726                          TST     (SP)+           ;ADJUST STACK
10931  054366  052737  000040  003052          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10932                                                                  ;  DRIVE ATTENTIONS
10933  054374  004737  052540                  JSR     PC,R.CONT       ;REPORT ERROR
10934  054400  000415                          BR      C.RTRN          ;RESTORE REGISTER
10935
10936  054402  005726                  39$:    TST     (SP)+           ;ADJUST STACK
10937  054404  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10938  054412  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
10939  054414  112737  177777  003074          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
10940  054422  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10941  054430  004737  052526          40$:    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
10942
10943  054434  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
10944  054440  012600                          MOV     (SP)+,R0        ;RESTORE R0
10945  054442  012601                          MOV     (SP)+,R1        ;RESTORE R1
10946  054444  012602                          MOV     (SP)+,R2        ;RESTORE R2
10947  054446  012603                          MOV     (SP)+,R3        ;RESTORE R3
10948  054450  012604                          MOV     (SP)+,R4        ;RESTORE R4
10949  054452  012605                          MOV     (SP)+,R5        ;RESTORE R5
10950  054454  000207                          RTS     PC              ;RETURN
10951                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
10952
10953                                  ;;*****************************************************************
10954                                  ;*
10955                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10956                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
10957                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10958                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
10959                                  ;*
10960                                  ;*CALL
10961                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
10962                                  ;*      JSR     PC,OCTBIN
10963                                  ;*      <ADDRESS OF ERROR RETURN>
10964                                  ;*      RETURN
10965                                  ;*
10966                                  ;;*****************************************************************
10967
10968  054456  010046          OCTBIN: MOV     R0,-(SP)        ;SAVE R0
10969  054460  010146                  MOV     R1,-(SP)        ;SAVE R1
10970  054462  010246                  MOV     R2,-(SP)        ;SAVE R2
10971  054464  016600  000010          MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
10972  054470  005001                  CLR     R1              ;CLEAR DATA WORDS
10973  054472  005002                  CLR     R2
10974  054474  112046          2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
10975  054476  001423                  BEQ     3$              ;IF ZERO GET OUT
10976  054500  121627  000054          CMPB    (SP),#',        ;CHECK IF COMMA
10977  054504  001420                  BEQ     3$              ;IF COMMA GET OUT
10978  054506  122716  000060          CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
10979  054512  003030                  BGT     4$              ;  AN OCTAL DIGIT
10980  054514  122716  000067          CMPB    #'7,(SP)
10981  054520  002425                  BLT     4$
```

```
 10982  054522  006301                        ASL     R1              ; *2
 10983  054524  006102                        ROL     R2
 10984  054526  006301                        ASL     R1              ; *4
 10985  054530  006102                        ROL     R2
 10986  054532  006301                        ASL     R1              ; *8
 10987  054534  006102                        ROL     R2
 10988  054536  042716  177770                BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
 10989  054542  062601                        ADD     (SP)+,R1        ;ADD THIS DIGIT
 10990  054544  000753                        BR      2$              ;LOOP
 10991  054546  005726                3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
 10992  054550  010166  000010                MOV     R1,10(SP)       ;SAVE RESULT
 10993  054554  010237  054610                MOV     R2,$HIOCT
 10994  054560  012602                        MOV     (SP)+,R2        ;RESTORE R2
 10995  054562  012601                        MOV     (SP)+,R1        ;RESTORE R1
 10996  054564  012600                        MOV     (SP)+,R0        ;RESTORE R0
 10997  054566  062716  000002                ADD     #2,(SP)         ;ADJUST RETURN
 10998  054572  000207                        RTS     PC              ;RETURN
 10999
 11000  054574  005726                4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
 11001  054576  012602                        MOV     (SP)+,R2        ;RESTORE R2
 11002  054600  012601                        MOV     (SP)+,R1        ;RESTORE R1
 11003  054602  012600                        MOV     (SP)+,R0        ;RESTORE R0
 11004  054604  013616                        MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
 11005  054606  000207                        RTS     PC              ;GO PROCESS ERROR
 11006  054610  000000                $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
 11007                                  .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
 11008
 11009                                  ;;**************************************************************************
 11010                                  ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
 11011                                  ;*WITH A RANGE OF 0 TO 2(+33)-1.
 11012                                  ;*CALL:
 11013                                  ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
 11014                                  ;*      RETURN                  ;;RETURN HERE THE RANDOM
 11015                                  ;*                              ;;NUMBER WILL BE IN
 11016                                  ;*                              ;;$HINUM,$LONUM
 11017
 11018  054612                  $RAND:
 11019  054612  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 11020  054614  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 11021  054616  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 11022  054620  013700  054712                MOV     $LONUM,R0       ;;SET R0 WITH LOW
 11023  054624  013701  054710                MOV     $HINUM,R1       ;;SET R1 WITH HIGH
 11024  054630  012702  177771                MOV     #-7,R2          ;;SET SHIFT COUNT
 11025  054634  006300                1$:     ASL     R0              ;;SHIFT R0 LEFT AND
 11026  054636  006101                        ROL     R1              ;;ROTATE CARRY INTO R1 AND
 11027  054640  005202                        INC     R2              ;;CHECK FOR DONE
 11028  054642  001374                        BNE     1$              ;;CONTINUE SHIFT LOOP
 11029  054644  063700  054712                ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
 11030  054650  005501                        ADC     R1              ;;PROPOGATE CARRY
 11031  054652  063701  054710                ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
 11032  054656  062700  001057                ADD     #1057,R0        ;;ADD LOW CONSTANT
 11033  054662  005501                        ADC     R1              ;;PROPOGATE CARRY
 11034  054664  062701  047401                ADD     #47401,R1       ;;ADD HIGH CONSTANT
 11035  054670  010037  054712                MOV     R0,$LONUM       ;;SAVE R0
 11036  054674  010137  054710                MOV     R1,$HINUM       ;;SAVE R1
 11037  054700  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
11038  054702  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
11039  054704  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
11040  054706  000207                  RTS     PC              ;;RETURN
11041  054710  176543          $HINUM: .WORD   176543
11042  054712  123456          $LONUM: .WORD   123456
11043                           .SBTTL  TYPE ROUTINE
11044
11045                           ;;**************************************************************
11046                           ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
11047                           ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
11048                           ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
11049                           ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
11050                           ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
11051                           ;*
11052                           ;*CALL:
11053                           ;*1) USING A TRAP INSTRUCTION
11054                           ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
11055                           ;*OR
11056                           ;*      TYPE
11057                           ;*      MESADR
11058                           ;*
11059
11060  054714  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
11061  054720  100002                  BPL     1$              ;;BR IF YES
11062  054722  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
11063  054724  000430                  BR      3$              ;;LEAVE
11064  054726  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
11065  054730  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
11066  054734  122737  000001  001340  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11067  054742  001011                  BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
11068  054744  132737  000100  001341  BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
11069  054752  001405                  BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
11070  054754  010037  054764          MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
11071  054760  004737  057412          JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
11072  054764  000000          61$:    .WORD   0               ;;MESSAGE ADDRESS
11073  054766  132737  000040  001341  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
11074  054774  001003                  BNE     60$             ;;YES,SKIP TYPE OUT
11075  054776  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
11076  055000  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
11077  055002  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
11078  055004  012600          60$:    MOV     (SP)+,R0        ;;RESTORE R0
11079  055006  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
11080  055012  000002                  RTI                     ;;RETURN
11081  055014  122716  000011  4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
11082  055020  001430                  BEQ     8$
11083  055022  122716  000200          CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
11084  055026  001006                  BNE     5$
11085  055030  005726                  TST     (SP)+           ;;POP  <CR><LF> EQUIV
11086  055032  104401                  TYPE                    ;;TYPE A CR AND LF
11087  055034  001315                  $CRLF
11088  055036  105037  055172          CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
11089  055042  000755                  BR      2$              ;;GET NEXT CHARACTER
11090  055044  004737  055126  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
11091  055050  123726  001156  6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
11092  055054  001350                  BNE     2$              ;;IF NO GO GET NEXT CHAR.
11093  055056  013746  001154          MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
11094                                                           ;;AND THE NULL CHAR.
11095  055062  105366  000001          7$:     DECB    1(SP)    ;;DOES A NULL NEED TO BE TYPED?
11096  055066  002770                          BLT     6$       ;;BR IF NO--GO POP THE NULL OFF OF STACK
11097  055070  004737  055126                  JSR     PC,$TYPEC ;;GO TYPE A NULL
11098  055074  105337  055172                  DECB    $CHARCNT ;;DO NOT COUNT AS A COUNT
11099  055100  000770                          BR      7$       ;;LOOP
11100
11101                                  ;HORIZONTAL TAB PROCESSOR
11102
11103  055102  112716  000040          8$:     MOVB    #' ,(SP) ;;REPLACE TAB WITH SPACE
11104  055106  004737  055126          9$:     JSR     PC,$TYPEC ;;TYPE A SPACE
11105  055112  132737  000007  055172          BITB    #7,$CHARCNT ;;BRANCH IF NOT AT
11106  055120  001372                          BNE     9$       ;;TAB STOP
11107  055122  005726                          TST     (SP)+    ;;POP SPACE OFF STACK
11108  055124  000724                          BR      2$       ;;GET NEXT CHARACTER
11109  055126  105777  124016          $TYPEC: TSTB    @$TPS    ;;WAIT UNTIL PRINTER IS READY
11110  055132  100375                          BPL     $TYPEC
11111  055134  116677  000002  124010          MOVB    2(SP),@$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
11112  055142  122766  000015  000002          CMPB    #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
11113  055150  001003                          BNE     1$       ;;BRANCH IF NO
11114  055152  105037  055172                  CLRB    $CHARCNT ;;YES--CLEAR CHARACTER COUNT
11115  055156  000406                          BR      $TYPEX   ;;EXIT
11116  055160  122766  000012  000002  1$:     CMPB    #LF,2(SP) ;;IS CHARACTER A LINE FEED?
11117  055166  001402                          BEQ     $TYPEX   ;;BRANCH IF YES
11118  055170  105227                          INCB    (PC)+    ;;COUNT THE CHARACTER
11119  055172  000000          $CHARCNT:.WORD  0                ;;CHARACTER COUNT STORAGE
11120  055174  000207          $TYPEX: RTS     PC
11121
11122
11123
11124
11125                                  ;;*****************************************************************
11126                                  .SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
11127                                  ;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
11128                                  ;*      USES ALL REGISTERS (R0-R5)
11129                                  ;*
11130                                  ;*      ENTER WITH   JSR   PC,M.DPIM
11131                                  ;*              MULTIPLIER IN R2-R3
11132                                  ;*              MULTIPLICAND IN R4-R5
11133                                  ;*              PRODUCT RETURNED IN R0-R1-R2-R3
11134                                  ;*
11135                                  ;;*****************************************************************
11136  055176  005000          M.DPIM: CLR     R0               ;CLEAR HI ORDER WORDS
11137  055200  005001                  CLR     R1
11138  055202  012746  000041                  MOV     #41,-(SP) ;MOVE 33 (DEC) TO COUNTER
11139  055206  006000          M.DP01: ROR     R0
11140  055210  006001                  ROR     R1
11141  055212  006002                  ROR     R2               ;SHIFT TO ADD
11142  055214  006003                  ROR     R3
11143  055216  103003                  BCC     M.DP02           ;NO CARRY NO ADD
11144  055220  060501                  ADD     R5,R1
11145  055222  005500                  ADC     R0               ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
11146  055224  060400                  ADD     R4,R0            ;  PRODUCT
11147  055226  005316          M.DP02: DEC     @SP              ;DECREMENT COUNTER
11148  055230  001366                  BNE     M.DP01
11149  055232  005726                  TST     (SP)+            ;REMOVE THE COUNTER
```

```
11150  055234  000207                          RTS     PC
11151
11152
11153                                  ;;*****************************************************************
11154                                  .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
11155                                  ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
11156                                  ;*              USES ALL REGISTERS (R0-R5)
11157                                  ;*
11158                                  ;*              ENTER WITH  JSR   PC,M.DPID
11159                                  ;*                      DIVIDEND IN R0-R1-R2-R3
11160                                  ;*                      DIVISOR IN R4-R5
11161                                  ;*                      REMAINDER RETURNED IN R0-R1
11162                                  ;*                      QUOTIENT RETURNED IN R2-R3
11163                                  ;*
11164                                  ;;*****************************************************************
11165  055236  012746  000040         M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
11166  055242  010446                          MOV     R4,-(SP)        ;HI ORDER
11167  055244  010546                          MOV     R5,-(SP)        ;LO ORDER DIVISOR TO THE STACK
11168  055246  005466  000002                 NEG     2(SP)           ;FORM NEGATIVE
11169  055252  005416                          NEG     @SP             ;   VERSION OF THE DIVISOR
11170  055254  005666  000002                 SBC     2(SP)
11171  055260  061601                          ADD     @SP,R1
11172  055262  005500                          ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
11173  055264  066600  000002                 ADD     2(SP),R0
11174  055270  103445                          BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
11175  055272  005046                          CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
11176  055274  006103         M.DP40: ROL     R3
11177  055276  006102                 ROL     R2
11178  055300  006101                 ROL     R1
11179  055302  006100                 ROL     R0
11180  055304  005716                 TST     @SP             ;TEST "CARRY INDICATOR"
11181  055306  001410                 BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
11182  055310  005016                 CLR     @SP             ;CLEAR UP FOR NEXT TIME
11183  055312  066601  000002                 ADD     2(SP),R1
11184  055316  005500                 ADC     R0              ;ADD -(DIVISOR)
11185  055320  005516                 ADC     @SP     ; I     ;SET "CARRY"
11186  055322  066600  000004                 ADD     4(SP),R0 ;<
11187  055326  000404                 BR      M.DP42
11188  055330  060501         M.DP41: ADD     R5,R1
11189  055332  005500                 ADC     R0              ;ADD +(DIVISOR)
11190  055334  005516                 ADC     @SP     ; I     ;SET "CARRY"
11191  055336  060400                 ADD     R4,R0   ;<
11192  055340  005516         M.DP42: ADC     @SP             ;SET "CARRY"
11193  055342  005716                 TST     @SP             ;TEST THE UPDATE INDICATOR
11194  055344  001401                 BEQ     .+4     ;>      ;IF ZERO FORGET IT
11195  055346  005203                 INC     R3      ; I     ;NO CARRY POSSIBLE HERE
11196  055350  005366  000006                 DEC     6(SP)   ;<      ;DECREMENT COUNTER
11197  055354  003347                 BGT     M.DP40          ;BR IF MORE TO DO
11198  055356  006003                 ROR     R3
11199  055360  103404                 BCS     M.DP44
11200  055362  060501                 ADD     R5,R1
11201  055364  005500                 ADC     R0
11202  055366  060400                 ADD     R4,R0
11203  055370  000241                 CLC
11204  055372  006103         M.DP44: ROL     R3
11205  055374  062706  000010                 ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
```

```
11206  055400  000242                          CLV
11207  055402  000207                          RTS     PC
11208  055404  062706  000006        M.DP50:   ADD     #6,SP
11209  055410  000262                          SEV
11210  055412  000207                          RTS     PC
11211
11212
11213
11214                                .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
11215
11216                                ;;**********************************************************************
11217                                ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
11218                                ;*UNSIGNED OCTAL ASCIZ NUMBER.
11219                                ;*CALL
11220                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
11221                                ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
11222                                ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
11223
11224
11225  055414  104407                $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
11226  055416  016601  000002                MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
11227  055422  012705  055533                MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
11228  055426  012704  000014                MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
11229  055432  012703  177770                MOV     #^C7,R3         ;;MASK
11230  055436  012100                        MOV     (R1)+,R0        ;;LOWER WORD
11231  055440  012101                        MOV     (R1)+,R1        ;;HIGH WORD
11232  055442  005002                        CLR     R2              ;;TERMINATOR
11233  055444  110245                1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
11234  055446  010002                        MOV     R0,R2           ;;GET THIS DIGIT
11235  055450  005304                        DEC     R4              ;;COUNT THIS CHARACTER
11236  055452  003007                        BGT     3$              ;;BR IF NOT THE LAST DIGIT
11237  055454  001405                        BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
11238  055456  005205                        INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
11239  055460  010566  000002                MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
11240  055464  104410                        RESREG                  ;;RESTORE ALL REGISTERS
11241  055466  000207                        RTS     PC              ;;RETURN TO USER
11242  055470  006203                2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
11243  055472  006001                3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
11244  055474  006000                        ROR     R0              ;;      THE NEXT OCTAL DIGIT
11245  055476  006001                        ROR     R1
11246  055500  006000                        ROR     R0
11247  055502  006001                        ROR     R1
11248  055504  006000                        ROR     R0
11249  055506  040302                        BIC     R3,R2           ;;MASK OUT ALL JUNK
11250  055510  062702  000060                ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
11251  055514  000753                        BR      1$              ;;GO PUT IT IN THE DATA TABLE
11252  055516  000016                $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
11253                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
11254
11255                                ;;**********************************************************************
11256                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
11257                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
11258                                ;*POSITIVE.
11259                                ;*CALL
11260                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
11261                                ;*      JSR     PC,@#$DB2D
```

```
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK
                                        $DB2D:  SAVREG                  ;;SAVE REGISTERS
055536  016602  000002                          MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
055542  012700  055714                          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
055546  010066  000002                          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
055552  012201                                  MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
055554  012202                                  MOV     (R2)+,R2
055556  012737  000012  055632                  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
055564  012704  055644                          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
055570  012705  055646                          MOV     #$TNPWR+2,R5
                                        1$:     CLR     R3              ;;CLEAR PARTIAL
055576  161401                          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
055600  005602                                  SBC     R2
055602  161502                                  SUB     (R5),R2
055604  002402                                  BLT     3$              ;;BR IF TEN POWER TO LARGE
055606  005203                                  INC     R3              ;;ADD 1 TO PARTIAL
055610  000772                                  BR      2$              ;;LOOP
055612  062401                          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
055614  005502                                  ADC     R2
055616  062402                                  ADD     (R4)+,R2
055620  022525                                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
055622  052703  000060                          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
055626  110320                                  MOVB    R3,(R0)+        ;;SAVE IT
055630  005327                                  DEC     (PC)+           ;;DONE?
055632  000000                          4$:     .WORD   0
055634  001357                                  BNE     1$              ;;BR IF NO
055636  105020                                  CLRB    (R0)+           ;;TERMINATOR
055640  104410                                  RESREG                  ;;RESTORE REGISTERS
055642  000207                                  RTS     PC              ;;RETURN
055644  145000                  $TNPWR: 145000                          ;;1.0E09
055646  035632                          35632
055650  160400                          160400                          ;;1.0E08
055652  002765                          2765
055654  113200                          113200                          ;;1.0E07
055656  000230                          230
055660  041100                          041100                          ;;1.0E06
055662  000017                          17
055664  103240                          103240                          ;;1.0E05
055666  000001                          1
055670  023420                          23420                           ;;1.0E04
055672  000000                          0
055674  001750                          1750                            ;;1.0E03
055676  000000                          0
055700  000144                          144                             ;;1.0E02
055702  000000                          0
055704  000012                          12                              ;;1.0E01
055706  000000                          0
055710  000001                          1                               ;;1.0E00
055712  000000                          0
055714  000014                  $DECVL: .BLKB   12.                     ;;RESERVE STORAGE FOR ASCIZ STRING
                                .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

```
11318                                   ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
11319                                   ;*LEADING NUMBERS.
11320                                   ;*CALL
11321                                   ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
11322                                   ;*      JSR     PC,@#$SUPRS
11323
11324
11325  055730  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
11326  055732  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
11327  055736  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
11328  055740  001403                          BEQ     2$              ;;BR IF YES
11329  055742  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
11330  055746  001773                          BEQ     1$              ;;BR IF YES
11331  055750  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
11332  055752  010037  055760                  MOV     R0,3$           ;;SAVE FOR TYPING
11333  055756  104401                          TYPE                    ;;GO TYPE
11334  055760  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
11335  055762  012600                          MOV     (SP)+,R0        ;;RESTORE R0
11336  055764  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
11337  055766  000207                          RTS     PC              ;;RETURN
11338                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11339
11340                                   ;;**********************************  **********************************
11341                                   ;*THIS ROUTINE IS USED TO CHANG_ A 16-BIT BINARY NUMBER TO A 6-DIGIT
11342                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11343                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11344                                   ;*CALL:
11345                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11346                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
11347                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11348                                   ;*      .BYTE   M               ;;M=1 OR 0
11349                                   ;*                                      ;;1=TYPE LEADING ZEROS
11350                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
11351                                   ;*
11352                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11353                                   ;*$TYPOS OR $TYPOC
11354                                   ;*CALL:
11355                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11356                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
11357                                   ;*
11358                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11359                                   ;*CALL:
11360                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11361                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
11362
11363  055770  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
11364  055774  116637  000001  056213          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
11365  056002  112637  056215                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
11366  056006  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
11367  056012  000406                          BR      $TYPON
11368  056014  112737  000001  056213  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
11369  056022  112737  000006  056215          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
11370  056030  112737  000005  056212  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
11371  056036  010346                          MOV     R3,-(SP)        ;;SAVE R3
11372  056040  010446                          MOV     R4,-(SP)        ;;SAVE R4
11373  056042  010546                          MOV     R5,-(SP)        ;;SAVE R5
```

```
11374  056044  113704  056215          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
11375  056050  005404                  NEG     R4
11376  056052  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11377  056056  110437  056214          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
11378  056062  113704  056213          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11379  056066  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11380  056072  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
11381  056074  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
11382  056076  000404                  BR      3$              ;;GO DO MSB
11383  056100  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
11384  056102  006105                  ROL     R5
11385  056104  006105                  ROL     R5
11386  056106  010503                  MOV     R5,R3
11387  056110  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
11388  056112  105337  056214          DECB    $OMODE          ;;TYPE THIS DIGIT?
11389  056116  100016                  BPL     7$              ;;BR IF NO
11390  056120  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
11391  056124  001002                  BNE     4$              ;;TEST FOR 0
11392  056126  005704                  TST     R4              ;;SUPPRESS THIS 0?
11393  056130  001403                  BEQ     5$              ;;BR IF YES
11394  056132  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
11395  056134  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11396  056140  052703  000040          BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
11397  056144  110337  056210          MOVB    R3,8$           ;;SAVE FOR TYPING
11398  056150  104401  056210          TYPE    ,8$             ;;GO TYPE THIS DIGIT
11399  056154  105337  056212  7$:     DECB    $OCNT           ;;COUNT BY 1
11400  056160  003347                  BGT     2$              ;;BR IF MORE TO DO
11401  056162  002402                  BLT     6$              ;;BR IF DONE
11402  056164  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11403  056166  000744                  BR      2$              ;;GO DO THE LAST DIGIT
11404  056170  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
11405  056172  012604                  MOV     (SP)+,R4        ;;RESTORE R4
11406  056174  012603                  MOV     (SP)+,R3        ;;RESTORE R3
11407  056176  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
11408  056204  012616                  MOV     (SP)+,(SP)
11409  056206  000002                  RTI                     ;;RETURN
11410  056210     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
11411  056211     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
11412  056212     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
11413  056213     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
11414  056214  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
11415                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11416
11417                          ;;**************************************************************
11418                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11419                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11420                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11421                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11422                          ;*REPLACED WITH SPACES.
11423                          ;*CALL:
11424                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
11425                          ;*      TYPDS                   ;;GO TO THE ROUTINE
11426
11427  056216                  $TYPDS:
11428  056216  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11429  056220  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
11430  056222  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11431  056224  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11432  056226  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11433  056230  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
11434  056234  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
11435  056240  100004                        BPL     1$              ;;BR IF INPUT IS POS.
11436  056242  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
11437  056244  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
11438  056252  005000                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
11439  056254  012703  056432                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
11440  056260  112723  000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
11441  056264  005002                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
11442  056266  016001  056422                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
11443  056272  160105                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
11444  056274  002402                        BLT     4$              ;;BR IF DONE
11445  056276  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
11446  056300  000774                        BR      3$
11447  056302  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
11448  056304  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
11449  056306  001002                        BNE     5$              ;;FALL THROUGH IF 0
11450  056310  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
11451  056312  100407                        BMI     7$              ;;BR IF YES
11452  056314  106316                5$:     ASLB    (SP)            ;;MSD?
11453  056316  103003                        BCC     6$              ;;BR IF NO
11454  056320  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
11455  056326  052702  000060          6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
11456  056332  052702  000040          7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11457  056336  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11458  056340  005720                        TST     (R0)+           ;;JUST INCREMENTING
11459  056342  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
11460  056346  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
11461  056350  003002                        BGT     8$              ;;GO TO EXIT
11462  056352  010502                        MOV     R5,R2           ;;GET THE LSD
11463  056354  000764                        BR      6$              ;;GO CHANGE TO ASCII
11464  056356  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
11465  056360  100003                        BPL     9$              ;;BR IF NO
11466  056362  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11467  056370  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
11468  056372  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
11469  056374  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
11470  056376  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
11471  056400  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
11472  056402  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
11473  056404  104401  056432                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
11474  056410  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11475  056416  012616                        MOV     (SP)+,(SP)
11476  056420  000002                        RTI                     ;;RETURN TO USER
11477  056422  023420                $DTBL:  10000.
11478  056424  001750                        1000.
11479  056426  000144                        100.
11480  056430  000012                        10.
11481  056432  000004                $DBLK:  .BLKW   4
11482                                 .SBTTL  ERROR HANDLER ROUTINE
11483
11484                                 ;;**************************************************************
11485                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

```
11486                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11487                                   ;*AND GO TO TYPERR ON ERROR
11488                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11489                                   ;*SW15=1        HALT ON ERROR
11490                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
11491                                   ;*SW10=1        BELL ON ERROR
11492                                   ;*SW09=1        LOOP ON ERROR
11493                                   ;*CALL
11494                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
11495
11496  056442                           $ERROR:
11497  056442  105237  001103           7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
11498  056446  001775                           BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
11499  056450  013777  001102  122464           MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
11500  056456  032777  002000  122454           BIT     #BIT10,@SWR     ;;BELL ON ERROR?
11501  056464  001402                           BEQ     1$              ;;NO - SKIP
11502  056466  104401  001310                   TYPE    ,$BELL          ;;RING BELL
11503  056472  005237  001112           1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
11504  056476  011637  001116                   MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
11505  056502  162737  000002  001116           SUB     #2,$ERRPC
11506  056510  117737  122402  001114           MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
11507  056516  032777  020000  122414           BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
11508  056524  001004                           BNE     20$             ;;SKIP TYPEOUTS
11509  056526  004737  042760                   JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
11510  056532  104401  001315                   TYPE    ,$CRLF
11511  056536                           20$:
11512  056536  122737  000001  001340           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11513  056544  001007                           BNE     2$              ;;NO,SKIP APT ERROR REPORT
11514  056546  113737  001114  056560           MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
11515  056554  004737  057422                   JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
11516  056560     000                   21$:    .BYTE   0
11517  056561     000                           .BYTE   0
11518  056562  000777                   22$:    BR      22$             ;;APT ERROR LOOP
11519  056564  005777  122350           2$:     TST     @SWR            ;;HALT ON ERROR
11520  056570  100001                           BPL     3$              ;;SKIP IF CONTINUE
11521  056572  000000                           HALT                    ;;HALT ON ERROR!
11522  056574  032777  001000  122336   3$.     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
11523  056602  001402                           BEQ     4$              ;;BR IF NO
11524  056604  013716  001110                   MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
11525  056610  005737  001306           4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
11526  056614  001402                           BEQ     5$              ;;BR IF NONE
11527  056616  013716  001306                   MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
11528  056622                           5$:
11529  056622  022737  024006  000042           CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
11530  056630  001001                           BNE     6$              ;;BRANCH IF NO
11531  056632  000000                           HALT                    ;;YES
11532  056634                           6$:
11533  056634  000002                           RTI                     ;;RETURN
11534                                   .SBTTL  TTY INPUT ROUTINE
11535
11536                                   ;;**************************************************************
11537                                   .ENABL  LSB
11538
11539                                   .DSABL  LSB
11540
11541
```

```
11542                                   ;;*****************************************************************
11543                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11544                                   ;*CALL:
11545                                   ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
11546                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
11547                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
11548                                   ;
11549
11550  056636  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
11551  056640  016666  000004  000002           MOV     4(SP),2(SP)     ;;SAVE THE PS
11552  056646  105777  122272          1$:     TSTB    @$TKS           ;;WAIT FOR
11553  056652  100375                           BPL     1$              ;;A CHARACTER
11554  056654  117766  122266  000004           MOVB    @$TKB,4(SP)     ;;READ THE TTY
11555  056662  042766  177600  000004           BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
11556  056670  026627  000004  000023           CMP     4(SP),#23       ;;IS IT A CONTROL-S?
11557  056676  001013                           BNE     3$              ;;BRANCH IF NO
11558  056700  105777  122240          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
11559  056704  100375                           BPL     2$              ;;LOOP UNTIL ITS THERE
11560  056706  117746  122234                   MOVB    @$TKB,-(SP)     ;;GET CHARACTER
11561  056712  042716  177600                   BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
11562  056716  022627  000021                   CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
11563  056722  001366                           BNE     2$              ;;IF NOT DISCARD IT
11564  056724  000750                           BR      1$              ;;YES, RESUME
11565  056726  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
11566  056734  002407                           BLT     4$              ;;BRANCH IF YES
11567  056736  026627  000004  000175           CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
11568  056744  003003                           BGT     4$              ;;BRANCH IF YES
11569  056746  042766  000040  000004           BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
11570  056754  000002                   4$:     RTI                     ;;GO BACK TO USER
11571  056756  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL ''U''
11572  056763     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL ''G''
11573  056770  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
11574  056776  020075     000
11575  057001     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
11576  057006  036440  000040
11577
11578
11579
11580                                   .SBTTL  POWER DOWN AND UP ROUTINES
11581
11582                                   ;POWER DOWN ROUTINE
11583  057012  012737  057024  000024  $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
11584  057020  000000                           HALT                    ;HANG UP
11585  057022  000776                           BR      .-2
11586
11587                                   ;POWER UP ROUTINE
11588  057024  005037  057076          $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
11589  057030  005237  057076          4$:     INC     $PWRCT
11590  057034  001375                           BNE     4$
11591  057036  012737  057012  000024           MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
11592  057044  012737  000340  000026           MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11593  057052  012737  000340  000036           MOV     #PR7,TRAPVEC+2
11594  057060  012706  001100                   MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
11595  057064  104401  057100                   TYPE    ,PWRMSG         ;TYPE ''POWER FAILED''
11596  057070  000005                           RESET                   ;CLEAR THE UNIBUS
11597  057072  000177  122010                   JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
11598  057076  000000                    $PWRCT: .WORD   0
11599  057100  005015  047520  042527    PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
11600  057106  020122  040506  046111
11601  057114  042105  005015     000
11602          057122                            .EVEN
11603
11604
11605
11606                                    ;;*****************************************************************
11607                                    ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
11608                                    ;*CALLED BY "SCOPER"
11609                                    ;;*****************************************************************
11610  057122  105737  001103           SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
11611  057126  001406                            BEQ     6$              ;BR IF NOT
11612  057130  032777  001000  122002            BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
11613  057136  001402                            BEQ     6$              ;BR IF NOT
11614  057140  013716  001110                    MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
11615  057144  000002                    6$:     RTI                     ;RETURN
11616
11617
11618
11619                                    .SBTTL  SCOPE HANDLER ROUTINE
11620
11621                                    ;;*****************************************************************
11622                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11623                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11624                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11625                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11626                                    ;*SW14=1         LOOP ON TEST
11627                                    ;*SW11=1         INHIBIT ITERATIONS
11628                                    ;*SW09=1         LOOP ON ERROR
11629                                    ;*CALL
11630                                    ;*      SCOPE           ;;SCOPE=IOT
11631
11632  057146                            $SCOPE:
11633  057146  005737  001304                    TST     $TIMES          ;CHECK CURRENT ITERATION NUMBER
11634  057152  001404                            BEQ     $XTSTR          ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
11635  057154  032777  040000  121756    1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11636  057162  001101                            BNE     $OVER           ;;YES IF SW14=1
11637                                    ;#####START OF CODE FOR THE XOR TESTER#####
11638  057164  000416                    $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11639                                                                    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11640  057166  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11641  057172  012737  057212  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11642  057200  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
11643  057204  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11644  057210  000450                            BR      $SVLAD          ;;GO TO THE NEXT TEST
11645  057212  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11646  057214  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11647  057220  000413                            BR      7$              ;;LOOP ON THE PRESENT TEST
11648  057222                            6$:;#####END OF CODE FOR THE XOR TESTER#####
11649  057222  105737  001103            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11650  057226  001421                            BEQ     3$              ;;BR IF NO
11651  057230  123737  001115  001103            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
11652  057236  101015                            BHI     3$              ;;BR IF NO
11653  057240  032777  001000  121672            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
```

```
11654  057246  001404                          BEQ     4$                ;;BR IF NO
11655  057250  013737  001110  001106  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
11656  057256  000443                          BR      $OVER
11657  057260  105037  001103          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
11658  057264  005037  001304                  CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11659  057270  000412                          BR      1$                ;;ESCAPE TO THE NEXT TEST
11660  057272  032777  004000  121640  3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
11661  057300  001006                          BNE     1$                ;;BR IF YES
11662  057302  005237  001104                  INC     $ICNT             ;;INCREMENT ITERATION COUNT
11663  057306  023737  001304  001104          CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
11664  057314  002024                          BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
11665  057316  012737  000001  001104  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
11666  057324  013737  057402  001304          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
11667  057332  105237  001102          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
11668  057336  113737  001102  001324          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
11669  057344  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
11670  057350  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
11671  057354  005037  001306                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11672  057360  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11673  057366  013777  001102  121546  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
11674  057374  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
11675  057400  000002                          RTI                       ;;FIXES PS
11676  057402  003720                  $MXCNT: 2000.                     ;;MAX. NUMBER OF ITERATIONS
11677                                  .SBTTL  APT COMMUNICATIONS ROUTINE
11678
11679                                  ;;*****************************************************************
11680  057404  112737  000001  057650  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
11681  057412  112737  000001  057646  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
11682  057420  000403                          BR      $ATYC
11683  057422  112737  000001  057650  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
11684  057430                          $ATYC:
11685  057430  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
11686  057432  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
11687  057434  105737  057646                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
11688  057440  001450                          BEQ     5$                ;;IF NOT:  BR
11689  057442  122737  000001  001340          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
11690  057450  001031                          BNE     3$                ;;IF NOT:  BR
11691  057452  132737  000100  001341          BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
11692  057460  001425                          BEQ     3$                ;;IF NOT:  BR
11693  057462  017600  000004                  MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
11694  057466  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
11695  057474  005737  001320          1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
11696  057500  001375                          BNE     1$                ;;IF NOT:  WAIT
11697  057502  010037  001334                  MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
11698  057506  105720                  2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
11699  057510  001376                          BNE     2$
11700  057512  163700  001334                  SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
11701  057516  006200                          ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
11702  057520  010037  001336                  MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
11703  057524  012737  000004  001320          MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
11704  057532  000413                          BR      5$
11705  057534  017637  000004  057560  3$:     MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
11706  057542  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
11707  057550  013746  177776                  MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
11708  057554  004737  054714                  JSR     PC,$TYPE          ;;CALL TYPE MACRO
11709  057560  000000                  4$:     .WORD   0
```

```
11710  057562                          5$:
11711  057562  105737  057650          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
11712  057566  001416                          BEQ     12$             ;;IF NOT:  BR
11713  057570  005737  001340                  TST     $ENV            ;;RUNNING UNDER APT?
11714  057574  001413                          BEQ     12$             ;;IF NOT:  BR
11715  057576  005737  001320          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
11716  057602  001375                          BNE     11$             ;;IF NOT:  WAIT
11717  057604  017637  000004  001322          MOV     @4(SP),$FATAL   ;;GET ERROR #
11718  057612  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
11719  057620  005237  001320                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
11720  057624  105037  057650          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
11721  057630  105037  057647                  CLRB    $LFLG           ;;CLEAR LOG FLAG
11722  057634  105037  057646                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
11723  057640  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11724  057642  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11725  057644  000207                          RTS     PC              ;;RETURN
11726  057646     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
11727  057647     000          $LFLG:  .BYTE   0               ;;LOG FLAG
11728  057650     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
11729          057652                  .EVEN
11730          000200          APTSIZE=200
11731          000001          APTENV=001
11732          000100          APTSPOOL=100
11733          000040          APTCSUP=040
11734                          .SBTTL  ROUTINE TO SIZE MEMORY
11735
11736                          ;;**************************************************************
11737                          ;*CALL:
11738                          ;*      JSR     PC,$SIZE
11739                          ;*      RETURN
11740                          ;*$LSTAD WILL CONTAIN:
11741                          ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11742                          ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11743                          ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11744                          ;*$KT11 IS THE MEMORY MANAGEMENT KEY
11745                          ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11746                          ;*      MUST BE SETUP BEFORE THE CALL
11747                          ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11748                          ;*      DETERMINED BY ROUTINE
11749
11750  057652  010046          $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
11751  057654  010146                  MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11752  057656  010246                  MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
11753  057660  010346                  MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
11754  057662  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
11755  057666  013746  000006          MOV     @#ERRVEC+2,-(SP)
11756  057672  010600                  MOV     SP,R0           ;;SAVE THE STACK POINTER
11757                          ;;SET THE ERRVEC PS TO THE PRESENT PS
11758  057674  104400                  TRAP                    ;;PUSH OLD PSW AND PC ON STACK
11759  057676  012637  000006          MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
11760  057702  012701  003776          MOV     #3776,R1        ;;SETUP ADDRESS
11761  057706  105727                  TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
11762  057710  000200          $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
11763  057712  100062                  BPL     $CORE           ;;BR IF NO
11764  057714  012737  060052  000004  MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
11765  057722  005737  177572          TST     @#SR0           ;;KT11 ARE YOU THERE?
```

```
11766  057726  052737  100000  057710          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
11767  057734  005046                          CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
11768  057736  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
11769  057742  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
11770  057746  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
11771  057754  011622                          MOV     (SP),(R2)+      ;;LOAD "PAR"
11772  057756  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
11773  057762  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
11774  057764  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
11775  057770  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
11776  057772  012737  060010  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
11777  060000  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
11778  060006  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
11779  060010  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
11780  060012  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
11781  060016  012737  060042  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
11782  060024  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
11783  060030  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
11784  060034  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
11785  060040  101371                          BHI     4$              ;;NO--TRY IT
11786  060042  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
11787  060044  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
11788  060050  000421                          BR      $SIZEX
11789  060052  042737  100000  057710  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
11790  060060  012737  060110  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
11791  060066  005002                          CLR     R2              ;;SET UP BANK
11792  060070  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
11793  060074  062702  000040                  ADD     #40,R2          ;;1K STEP
11794  060100  005711                          TST     (R1)            ;;TRAP ON TIME OUT
11795  060102  022701  177776                  CMP     #177776,R1      ;;LAST ONE
11796  060106  001370                          BNE     1$              ;;NO--TRY AGAIN
11797  060110  162701  004000          $CROUT: SUB     #4000,R1
11798  060114  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
11799  060120  010006                          MOV     R0,SP           ;;RESTORE THE STACK
11800  060122  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
11801  060126  012637  000004                  MOV     (SP)+,@#ERRVEC
11802  060132  010137  060154                  MOV     R1,$LSTAD       ;;LAST ADDRESS
11803  060136  010237  060156                  MOV     R2,$LSTBK       ;;LAST BANK
11804  060142  012603                          MOV     (SP)+,R3        ;;RESTORE R3
11805  060144  012602                          MOV     (SP)+,R2        ;;RESTORE R2
11806  060146  012601                          MOV     (SP)+,R1        ;;RESTORE R1
11807  060150  012600                          MOV     (SP)+,R0        ;;RESTORE R0
11808  060152  000207                          RTS     PC
11809  060154  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
11810  060156  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
11811                           .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11812
11813                           ;;**************************************************************
11814                           ;*SAVE R0-R5
11815                           ;*CALL:
11816                           ;*      SAVREG
11817                           ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11818                           ;*
11819                           ;*TOP---(+16)
11820                           ;* +2---(+18)
11821                           ;* +4---R5
```

```
11822                                   ;* +6---R4
11823                                   ;* +8---R3
11824                                   ;*+10---R2
11825                                   ;*+12---R1
11826                                   ;*+14---R0
11827
11828  060160                           $SAVREG:
11829  060160  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11830  060162  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11831  060164  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11832  060166  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11833  060170  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11834  060172  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11835  060174  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11836  060200  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11837  060204  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11838  060210  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11839  060214  000002                          RTI
11840
11841                                   ;*RESTORE R0-R5
11842                                   ;*CALL:
11843                                   ;*      RESREG
11844  060216                           $RESREG:
11845  060216  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11846  060222  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11847  060226  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11848  060232  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11849  060236  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11850  060240  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11851  060242  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11852  060244  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11853  060246  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11854  060250  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11855  060252  000002                          RTI
11856                           .SBTTL  TRAP DECODER
11857
11858                           ;;*************************************************************
11859                           ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11860                           ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11861                           ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11862                           ;*GO TO THAT ROUTINE.
11863
11864  060254  010046           $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
11865  060256  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
11866  060262  005740                  TST     -(R0)           ;;BACKUP BY 2
11867  060264  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
11868  060266  006300                  ASL     R0              ;;POSITION FOR INDEXING
11869  060270  016000  060310          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
11870  060274  000200                  RTS     R0              ;;GO TO ROUTINE
11871
11872
11873                           ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
11874
11875  060276  011646           $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
11876  060300  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
11877  060306  000002                  RTI                     ;;RESTORE THE PSW
```

```
11878
11879                                   .SBTTL  TRAP TABLE
11880
11881                                   ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
11882                                   ;*BY THE "TRAP" INSTRUCTION.
11883
11884                                   ;       ROUTINE
11885                                   ;       -------
11886  060310  060276                  $TRPAD: .WORD   $TRAP2
11887  060312  054714                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
11888  060314  056014                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
11889  060316  055770                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
11890  060320  056030                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
11891  060322  056216                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
11892
11893
11894  060324  056636                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
11895  060326  060160                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
11896  060330  060216                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
11897  060332  057122                          SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
11898
11899  060334  020040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
11900  060342  052123  000040
11901  060346  025052  020040     000  AS2SP2: .ASCIZ  /**  /
11902  060353     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PAR ERR/
11903  060360  020123  040520  020122
11904  060366  051105  000122
11905  060372  047516  026516  054105  EM2:    .ASCIZ  /NON-EXIST MEM/
11906  060400  051511  020124  042515
11907  060406  000115
11908  060410  047516  026516  054105  EM3:    .ASCIZ  /NON-EXIST DRV/
11909  060416  051511  020124  051104
11910  060424  000126
11911  060426  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERR/
11912  060434  042511  042114  042440
11913  060442  051122     000
11914  060445     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
11915  060452  020123  044524  042515
11916  060460  052517  000124
11917  060464  020104  047524  041440  EM6:    .ASCIZ  /D TO C PAR ERR/
11918  060472  050040  051101  042440
11919  060500  051122     000
11920  060503     103  052040  020117  EM7:    .ASCIZ  /C TO D PAR ERR/
11921  060510  020104  040520  020122
11922  060516  051105  000122
11923  060522  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
11924  060530     000
11925  060531     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
11926  060536  046040  051517  000123
11927  060544  046111  020114  052506  EM12:   .ASCIZ  /ILL FUNCT/
11928  060552  041516  000124
11929  060556  051120  043517  042440  EM13:   .ASCIZ  /PROG ERR/
11930  060564  051122     000
11931  060567     116  047117  042455  EM14:   .ASCIZ  /NON-EXIST FUNCT/
11932  060574  044530  052123  043040
11933  060602  047125  052103     000
```

```
11934  060607     104  053122  052040  EM15:   .ASCIZ  /DRV TYP ERR/
11935  060614  050131  042440  051122
11936  060622     000
11937  060623     106  040515  020124  EM16:   .ASCIZ  /FMAT ERR/
11938  060630  051105  000122
11939  060634  051127  020124  047514  EM17:   .ASCIZ  /WRT LOCK ERR/
11940  060642  045503  042440  051122
11941  060650     000
11942  060651     104  053122  052440  EM20:   .ASCIZ  /DRV UNSAFE/
11943  060656  051516  043101  000105
11944  060664  042523  045505  044440  EM21:   .ASCIZ  /SEEK INCOMP/
11945  060672  041516  046517  000120
11946  060700  054503  020114  053117  EM22:   .ASCIZ  /CYL OVRFLO/
11947  060706  043122  047514     000
11948  060713     111  046114  041440  EM23:   .ASCIZ  /ILL CYL ADDR/
11949  060720  046131  040440  042104
11950  060726  000122
11951  060730  051104  020126  043117  EM24:   .ASCIZ  /DRV OFF TRACK/
11952  060736  020106  051124  041501
11953  060744  000113
11954  060746  051104  020126  044524  EM25:   .ASCIZ  /DRV TIMING ERR/
11955  060754  044515  043516  042440
11956  060762  051122     000
11957  060765     104  052101  020101  EM26:   .ASCIZ  /DATA LATE/
11958  060772  040514  042524     000
11959  060777     103  047117  051124  EM27:   .ASCIZ  /CONTR TIMEOUT/
11960  061004  052040  046511  047505
11961  061012  052125     000
11962  061015     117  042520  040522  EM30:   .ASCIZ  /OPERATION INCOMP/
11963  061022  044524  047117  044440
11964  061030  041516  046517  000120
11965  061036  042510  042101  051105  EM31:   .ASCIZ  /HEADER VRC ERR/
11966  061044  053040  041522  042440
11967  061052  051122     000
11968  061055     104  052101  020101  EM32:   .ASCIZ  /DATA CHECK ERR/
11969  061062  044103  041505  020113
11970  061070  051105  000122
11971  061074  051127  020124  044103  EM33:   .ASCIZ  /WRT CHK ERR/
11972  061102  020113  051105  000122
11973  061110  040504  040524  046440  EM34:   .ASCIZ  /DATA MISCOMPARE/
11974  061116  051511  047503  050115
11975  061124  051101  000105
11976  061130  047516  042040  053122  EM35:   .ASCIZ  /NO DRV RESPONSE-UFE & NXD/
11977  061136  051040  051505  047520
11978  061144  051516  026505  043125
11979  061152  020105  020046  054116
11980  061160  000104
11981  061162  051104  020126  051105  EM36:   .ASCIZ  /DRV ERR WILL NOT CLEAR/
11982  061170  020122  044527  046114
11983  061176  047040  052117  041440
11984  061204  042514  051101     000
11985  061211     104  053122  051440  EM37:   .ASCIZ  /DRV STATUS CHANGE WILL NOT CLEAR/
11986  061216  040524  052524  020123
11987  061224  044103  047101  042507
11988  061232  053440  046111  020114
11989  061240  047516  020124  046103
```

```
 11990  061246  040505  000122
 11991  061252  052101  023524  020116  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
 11992  061260  052502  020124  047516
 11993  061266  051440  040524  052524
 11994  061274  020123  044103  047101
 11995  061302  042507  047440  020122
 11996  061310  040506  046125  000124
 11997  061316  052101  023524  020116  EM41:   .ASCIZ  /ATT'N BUT DRV NOT AVAIL/
 11998  061324  052502  020124  051104
 11999  061332  020126  047516  020124
 12000  061340  053101  044501  000114
 12001  061346  052101  023524  020116  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
 12002  061354  044127  047105  047040
 12003  061362  052117  042440  050130
 12004  061370  041505  042524  000104
 12005  061376  051105  020122  040507  EM43:   .ASCIZ  /ERR GATHERING DRV STATUS/
 12006  061404  044124  051105  047111
 12007  061412  020107  051104  020126
 12008  061420  052123  052101  051525
 12009  061426     000
 12010  061427     115  046125  020124  EM52:   .ASCIZ  /MULT DRV SEL/
 12011  061434  051104  020126  042523
 12012  061442  000114
 12013  061444  042510  042101  051105  EM53:   .ASCIZ  /HEADER COMPARE ERR/
 12014  061452  041440  046517  040520
 12015  061460  042522  042440  051122
 12016  061466     000
 12017  061467     123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
 12018  061474  020123  044524  042515
 12019  061502  052517  000124
 12020  061506  051105  020122  047111  EM60:   .ASCIZ  /ERR IN RECAL FOR RECOVERY/
 12021  061514  051040  041505  046101
 12022  061522  043040  051117  051040
 12023  061530  041505  053117  051105
 12024  061536  000131
 12025  061540  051104  050117  044520  EM61:   .ASCIZ  /DROPPING DRIVE FATAL ERROR/
 12026  061546  043516  042040  044522
 12027  061554  042526  043040  052101
 12028  061562  046101  042440  051122
 12029  061570  051117     000
 12030  061573     103  046131  046440  EM62:   .ASCIZ  /CYL MISCOMP/
 12031  061600  051511  047503  050115
 12032  061606     000
 12033  061607     103  042514  051101  EM63:   .ASCIZ  /CLEAR CONTR DID NOT CLEAR ERR/
 12034  061614  041440  047117  051124
 12035  061622  042040  042111  047040
 12036  061630  052117  041440  042514
 12037  061636  051101  042440  051122
 12038  061644     000
 12039  061645     116  020117  052101  EM64:   .ASCIZ  /NO ATT'N IN ATT'N REG/
 12040  061652  023524  020116  047111
 12041  061660  040440  052124  047047
 12042  061666  051040  043505     000
 12043  061673     125  051516  046117  EM65:   .ASCIZ  /UNSOLICITED ATT'N/
 12044  061700  041511  052111  042105
 12045  061706  040440  052124  047047
```

```
12046  061714     000
12047  061715     125  042516  050130  EM66:   .ASCIZ  /UNEXPECTED DATA ERR/
12048  061722  041505  042524  020104
12049  061730  040504  040524  042440
12050  061736  051122     000
12051  061741     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
12052  061746  042040  042111  047040
12053  061754  052117  051040  051505
12054  061762  052105  053440  052111
12055  061770  020110  046103  040505
12056  061776  000122
12057  062000  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRV ATT'N/
12058  062006  041440  042514  051101
12059  062014  042040  042111  047040
12060  062022  052117  041440  042514
12061  062030  051101  042040  053122
12062  062036  040440  052124  047047
12063  062044     000
12064  062045     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/
12065  062052  040514  042524  053440
12066  062060  042510  020116  047125
12067  062066  047514  042101  047111
12068  062074  020107  042510  042101
12069  062102  051105     000
12070  062105     103  047117  051124  EM72:   .ASCIZ  /CONTR ERR WHEN DRIVER SERV/
12071  062112  042440  051122  053440
12072  062120  042510  020116  051104
12073  062126  053111  051105  051440
12074  062134  051105  000126
12075  062140  051104  020126  042504  EM73:   .ASCIZ  /DRV DET PAR ERR/
12076  062146  020124  040520  020122
12077  062154  051105  000122
12078  062160  047125  042504  020106  EM74:   .ASCIZ  /UNDEF ERR/
12079  062166  051105  000122
12080  062172  040515  045522  047111  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
12081  062200  020107  044124  051511
12082  062206  051440  041505  047524
12083  062214  020122  040502  000104
12084  062222  040502  020104  040504  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
12085  062230  040524  053040  051105
12086  062236  043111  047047  053440
12087  062244  052111  020110  042522
12088  062252  042101  020056  041505
12089  062260  020103  043117  046040
12090  062266  051501  020124  042522
12091  062274  051124  020131  051511
12092  062302  000072
12093  062304  042522  051124  020131  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
12094  062312  052523  041503  051505
12095  062320  043123  046125     000
12096  062325     122  052105  054522  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
12097  062332  052440  051516  041525
12098  062340  042503  051523  052506
12099  062346  000114
12100  062350  040503  047116  052117  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
12101  062356  043040  047111  020104
```

```
12102  062364  020101  040526  044514
12103  062372  020104  042510  042101
12104  062400  051105  044440  020116
12105  062406  051124  041501  020113
12106  062414  052512  052123  051040
12107  062422  040505  000104
12108  062426  040502  020104  042523  EM102:  .ASCIZ  /BAD SECT ERR ON SECT NOT LISTED BAD/
12109  062434  052103  042440  051122
12110  062442  047440  020116  042523
12111  062450  052103  047040  052117
12112  062456  046040  051511  042524
12113  062464  020104  040502  000104
12114  062472  044524  042515  026504  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
12115  062500  052517  020124  047117
12116  062506  051040  040505  020104
12117  062514  042110  000122
12118  062520  044524  042515  026504  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
12119  062526  052517  020124  047117
12120  062534  051440  042505  000113
12121  062542  051104  020126  044523  EM105:  .ASCIZ  /DRV SIEZED BY OTHER PORT/
12122  062550  055105  042105  041040
12123  062556  020131  052117  042510
12124  062564  020122  047520  052122
12125  062572     000
12126  062573     104  052101  020101  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
12127  062600  044515  041523  050115
12128  062606  020122  044127  046111
12129  062614  020105  040502  020111
12130  062622  042523  000124
12131  062626  047516  047040  046505  EM107:  .ASCIZ  /NO NEM ERR WHEN REF'ING LOC 760000/
12132  062634  042440  051122  053440
12133  062642  042510  020116  042522
12134  062650  023506  047111  020107
12135  062656  047514  020103  033067
12136  062664  030060  030060     000
12137  062671     111  052116  050122  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
12138  062676  020124  044127  047105
12139  062704  041440  052116  046122
12140  062712  020122  047516  020124
12141  062720  042122  000131
12142  062724  047516  040440  052124  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
12143  062732  047047  047440  020116
12144  062740  042523  045505     000
12145  062745     104  053122  051447  EM112:  .ASCIZ  /DRV'S CYL INCORRECT/
12146  062752  041440  046131  044440
12147  062760  041516  051117  042522
12148  062766  052103     000
12149  062771     101  047502  052122  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
12150  062776  020055  040503  023516
12151  063004  020124  042522  042101
12152  063012  041040  043123     000
12153  063017     113  030524  020061  EM114:  .ASCIZ  /KT11 FAILURE/
12154  063024  040506  046111  051125
12155  063032  000105
12156  063034  042515  020115  040520  EM115:  .ASCIZ  /MEM PAR ERR/
12157  063042  020122  051105  000122
```

```
12158  063050  052503  051122  047105  DH100:  .ASCIZ  /CURRENT CMD :/
12159  063056  020124  046503  020104
12160  063064  000072
12161  063066  051120  053105  041440  DH105:  .ASCIZ  /PREV CMD :/
12162  063074  042115  035040     000
12163  063101     122  033113  030461  DH200:  .ASCIZ  /RK611 REGS :/
12164  063106  051040  043505  020123
12165  063114  000072
12166  063116  042522  040515  047111  DH500:  .ASCIZ  /REMAINING REGS. NOT VALID/
12167  063124  047111  020107  042522
12168  063132  051507  047040  052117
12169  063140  053040  046101  042111
12170  063146     000
12171  063147     124  042510  043040  DH501:  .ASCIZ  /THE FOLLOWING REG DATA MAY BE INCORRECT :/
12172  063154  046117  047514  044527
12173  063162  043516  051040  043505
12174  063170  042040  052101  020101
12175  063176  040515  020131  042502
12176  063204  044440  041516  051117
12177  063212  042522  052103  035040
12178  063220     000
12179  063221     105  051122  050040  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT/
12180  063226  020103  042040  044522
12181  063234  042526  020040  041440
12182  063242  047115  020104  020040
12183  063250  041440  046131  042116
12184  063256  020122  052040  040522
12185  063264  045503  020040  051440
12186  063272  041505  047524  020122
12187  063300  053440  020104  047103
12188  063306  000124
12189  063310  044510  041040  020101  DH102:  .ASCIZ  /HI BA   LO BA/
12190  063316  020040  047514  041040
12191  063324  000101
12192  063326  051120  053105  020056  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
12193  063334  047125  041111  051525
12194  063342  046440  050101  035040
12195  063350     000
12196  063351     110  020111  042522  DH104:  .ASCIZ  /HI REGO LO REGO/
12197  063356  030107  046040  020117
12198  063364  042522  030107     000
12199  063371     103  051125  020056  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
12200  063376  047125  041111  051525
12201  063404  046440  050101  035040
12202  063412     000
12203  063413     122  041513  030523  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
12204  063420  020040  051040  041513
12205  063426  031123  020040  051040
12206  063434  042113  020103  020040
12207  063442  051040  042113  020101
12208  063450  020040  051040  053513
12209  063456  051103  020040  051040
12210  063464  041113  020101  020040
12211  063472  051040  040513  047523
12212  063500  000106
12213  063502  045522  051504  020040  DH202:  .ASCIZ  /RKDS    RKER/
```

```
12214  063510  020040  045522  051105
12215  063516     000
12216  063517     101  030060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
12217  063524  020040  041040  030060
12218  063532  020040  020040  040440
12219  063540  030460  020040  020040
12220  063546  041040  030460  020040
12221  063554  020040  040440  031060
12222  063562  020040  020040  041040
12223  063570  031060  020040  020040
12224  063576  040440  031460  020040
12225  063604  020040  041040  031460
12226  063612     000
12227  063613     110  040505  042504  DH601:  .ASCIZ  /HEADER SHOULD BE:/
12228  063620  020122  044123  052517
12229  063626  042114  041040  035105
12230  063634     000
12231  063635     110  040505  042504  DH602:  .ASCIZ  /HEADER OF SECT 0 THIS CYL IS:/
12232  063642  020122  043117  051440
12233  063650  041505  020124  020060
12234  063656  044124  051511  041440
12235  063664  046131  044440  035123
12236  063672     000
12237  063673     120  041501  020113  DH604:  .ASCIZ  /PACK ADDR OF ERR(S) :/
12238  063700  042101  051104  047440
12239  063706  020106  051105  024122
12240  063714  024523  035040     000
12241  063721     103  046131  042116  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
12242  063726  020122  052040  040522
12243  063734  045503  020040  051440
12244  063742  041505  047524  000122
12245  063750  054503  047114  051104  DH6042: .ASCIZ  /CYLNDR  TRACK/
12246  063756  020040  051124  041501
12247  063764  000113
12248  063766  041505  020103  040504  DH605:  .ASCIZ  /ECC DATA IS:/
12249  063774  040524  044440  035123
12250  064002     000
12251  064003     120  051517  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
12252  064010  020040  050040  052101
12253  064016  020040  020040  041440
12254  064024  051117  042522  052103
12255  064032  041101  042514  000077
12256  064040  047527  042122  020061  DH606:  .ASCIZ  /WORD1   WORD2   WORD3/
12257  064046  020040  047527  042122
12258  064054  020062  020040  047527
12259  064062  042122  000063
12260  064066  040502  020104  042510  DH607:  .ASCIZ  /BAD HEADER IS :/
12261  064074  042101  051105  044440
12262  064102  020123  000072
12263  064106  042127  021440  020040  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
12264  064114  020040  047507  042117
12265  064122  020040  020040  040502
12266  064130  020104  020040  020040
12267  064136  044510  050040  054510
12268  064144  020040  047514  050040
12269  064152  054510  020040  051126
```

```
12270  064160  020124  042101  020040
12271  064166  044513  040520  033122
12272  064174     000
12273  064175     107  047517  020104  DH702:  .ASCIZ  /GOOD    BAD/
12274  064202  020040  041040  042101
12275  064210     000
12276  064211     106  044501  044514  DH703:  .ASCIZ  /FAILING DATA WORD :/
12277  064216  043516  042040  052101
12278  064224  020101  047527  042122
12279  064232  035040     000
12280  064235     123  030122  051411  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
12281  064242  030522  051411  031122
12282  064250  051411  031522     000
12283  064255     113  030524  020061  DH705:  .ASCIZ  /KT11 REGS :/
12284  064262  042522  051507  035040
12285  064270     000
12286  064271     115  046505  051040  DH706:  .ASCIZ  /MEM REGS :/
12287  064276  043505  020123  000072
12288  064304  041520  046011  042517  DH707:  .ASCIZ  /PC     LOERAD  HIERAD  MEMSYS/
12289  064312  040522  004504  044510
12290  064320  051105  042101  046411
12291  064326  046505  054523  000123
12292  064334  041520  041411  051123  DH710:  .ASCIZ  /PC     CSR AD  CSR/
12293  064342  040440  004504  051503
12294  064350  000122
12295  064352  054503  044514  042116  DH711:  .ASCIZ  /CYLINDERS :/
12296  064360  051105  020123  000072
12297  064366  045522  051504  020040  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
12298  064374  020040  045522  051105
12299  064402  020040  020040  045522
12300  064410  051115  020062  020040
12301  064416  045522  051115  000063
12302  064424  051105  047522  020122  DH502:  .ASCIZ  /ERROR TRYING TO GET FAILURE INFO/
12303  064432  051124  044531  043516
12304  064440  052040  020117  042507
12305  064446  020124  040506  046111
12306  064454  051125  020105  047111
12307  064462  047506     000
12308  064465     123  041505  047117  DH503:  .ASCIZ  /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
12309  064472  020104  044524  042515
12310  064500  052517  020124  041517
12311  064506  052503  051122  042105
12312  064514  043440  052105  044524
12313  064522  043516  042040  044522
12314  064530  042526  051440  040524
12315  064536  052524  000123
12316  064542  052516  041115  051105  DH800:  .ASCIZ  /NUMBER OF RETRIES:/
12317  064550  047440  020106  042522
12318  064556  051124  042511  035123
12319  064564     000
12320          064566                          .EVEN
12321  064566  001116  005500  005476  DT100:  .WORD   $FRRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
12322  064574  001162  001164  001166
12323  064602  001170
12324  064604  001256  001260          DT102:  .WORD   $REG36,$REG37
12325  064610  001174  001176  001200  DT201:  .WORD   $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
```

```
12326  064616  001202  001204  001206
12327  064624  001210
12328  064626  001212  001214          DT202:  .WORD   $REG14,$REG15
12329  064632  001216  001220  001222  DT203:  .WORD   $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
12330  064640  001224  001226  001230
12331  064646  001232  001234
12332  064652  001174  001176  001200  DT601:  .WORD   $REG5,$REG6,$REG7
12333  064660  001202  001204  001206  DT602:  .WORD   $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
12334  064666  001210  001212  001214
12335  064674  001216
12336  064676  005256  005254          DT103:  .WORD   PRMPH0,PRMPL0
12337
12338  064702  000006                  DF01:   .WORD   6                ;NUMBER OF HEADER LINES
12339  064704     000                          .BYTE   0                ;NUMBER OF COL FOR FIRST HDR
12340  064705     000                          .BYTE   0                ;ALL CHARACTERS OCTAL
12341  064706  063221                          .WORD   DH101            ;SECOND HDR LINE
12342  064710     007     000                  .BYTE   7,0              ;NUM OF COL-ALL OCTAL
12343  064712  063310                          .WORD   DH102
12344  064714     002     000                  .BYTE   2,0
12345  064716  063101                          .WORD   DH200
12346  064720     000     000                  .BYTE   0,0
12347  064722  063413                          .WORD   DH201
12348  064724     007     000                  .BYTE   7,0
12349  064726  063116                          .WORD   DH500
12350  064730     000     000                  .BYTE   0,0
12351
12352  064732  000007                  DF02:   .WORD   7
12353  064734     000     000                  .BYTE   0,0
12354  064736  063221                          .WORD   DH101
12355  064740     007     000                  .BYTE   7,0
12356  064742  063310                          .WORD   DH102
12357  064744     002     000                  .BYTE   2,0
12358  064746  063101                          .WORD   DH200
12359  064750     000     000                  .BYTE   0,0
12360  064752  063413                          .WORD   DH201
12361  064754     007     000                  .BYTE   7,0
12362  064756  063502                          .WORD   DH202
12363  064760     002     000                  .BYTE   2,0
12364  064762  063517                          .WORD   DH203
12365  064764     010     000                  .BYTE   10,0
12366
12367  064766  000010                  DF03:   .WORD   10
12368  064770     000     000                  .BYTE   0,0
12369  064772  063221                          .WORD   DH101
12370  064774     007     000                  .BYTE   7,0
12371  064776  063310                          .WORD   DH102
12372  065000     002     000                  .BYTE   2,0
12373  065002  063101                          .WORD   DH200
12374  065004     000     000                  .BYTE   0,0
12375  065006  063413                          .WORD   DH201
12376  065010     007     000                  .BYTE   7,0
12377  065012  063147                          .WORD   DH501            ;''THE FOLLOWING REG DATA MB INCORRECT''
12378  065014     000     000                  .BYTE   0,0
12379  065016  063502                          .WORD   DH202
12380  065020     002     000                  .BYTE   2,0
12381  065022  063517                          .WORD   DH203
```

```
12382  065024     010     000                .BYTE   10,0
12383
12384  065026  000003                 DF04:  .WORD   3
12385  065030     000     000                .BYTE   0,0
12386  065032  063221                        .WORD   DH101
12387  065034     007     000                .BYTE   7,0
12388  065036  063310                        .WORD   DH102
12389  065040     002     000                .BYTE   2,0
12390
12391  065042  000007                 DF05:  .WORD   7                ;OPI, HVRC
12392  065044     000     000                .BYTE   0,0
12393  065046  063221                        .WORD   DH101
12394  065050     007     000                .BYTE   7,0
12395  065052  063310                        .WORD   DH102
12396  065054     002     000                .BYTE   2,0
12397  065056  063613                        .WORD   DH601
12398  065060     000     000                .BYTE   0,0
12399  065062  064040                        .WORD   DH606
12400  065064     003     000                .BYTE   3,0
12401  065066  063635                        .WORD   DH602
12402  065070     000     000                .BYTE   0,0
12403  065072  064040                        .WORD   DH606
12404  065074     003     000                .BYTE   3,0
12405
12406  065076  000007                 DF07:  .WORD   7
12407  065100     000     000                .BYTE   0,0
12408  065102  063221                        .WORD   DH101
12409  065104     007     000                .BYTE   7,0
12410  065106  063310                        .WORD   DH102
12411  065110     002     000                .BYTE   2,0
12412  065112  063673                        .WORD   DH604
12413  065114     000     000                .BYTE   0,0
12414  065116  063721                        .WORD   DH6041
12415  065120     003     000                .BYTE   3,0
12416  065122  063766                        .WORD   DH605
12417  065124     000     000                .BYTE   0,0
12418  065126  064003                        .WORD   DH6051
12419  065130     003     000                .BYTE   3,0
12420
12421  065132  000005                 DF10:  .WORD   5
12422  065134     000     000                .BYTE   0,0
12423  065136  063221                        .WORD   DH101
12424  065140     007     000                .BYTE   7,0
12425  065142  063310                        .WORD   DH102
12426  065144     002     000                .BYTE   2,0
12427  065146  063673                        .WORD   DH604
12428  065150     000     000                .BYTE   0,0
12429  065152  063721                        .WORD   DH6041
12430  065154     003     000                .BYTE   3,0
12431
12432  065156  000004                 DF11:  .WORD   4
12433  065160     000     000                .BYTE   0,0
12434  065162  063673                        .WORD   DH604
12435  065164     000     000                .BYTE   0,0
12436  065166  063721                        .WORD   DH6041
12437  065170     003     000                .BYTE   3,0
```

```
12438  065172  064106                      .WORD   DH701
12439  065174     000      000             .BYTE   0,0
12440
12441  065176  000006              DF12:   .WORD   6
12442  065200     000      000             .BYTE   0,0
12443  065202  063221                      .WORD   DH101
12444  065204     007      000             .BYTE   7,0
12445  065206  063310                      .WORD   DH102
12446  065210     002      000             .BYTE   2,0
12447  065212  063101                      .WORD   DH200
12448  065214     000      000             .BYTE   0,0
12449  065216  063413                      .WORD   DH201
12450  065220     007      000             .BYTE   7,0
12451  065222  064366                      .WORD   DH204
12452  065224     004      000             .BYTE   4,0
12453
12454  065226  000006              DF13:   .WORD   6               ;FORMAT FOR 2ND LEVEL ERROR
12455  065230     000      000             .BYTE   0,0             ;IN HEADER COMPARE ERROR
12456  065232  063221                      .WORD   DH101           ;AND 2ND LEVEL HEADER
12457  065234     007      000             .BYTE   7,0             ;VRC ERROR
12458  065236  063310                      .WORD   DH102
12459  065240     002      000             .BYTE   2,0
12460  065242  063613                      .WORD   DH601
12461  065244     000      000             .BYTE   0,0
12462  065246  064040                      .WORD   DH606
12463  065250     003      000             .BYTE   3,0
12464  065252  064424                      .WORD   DH502
12465  065254     000      000             .BYTE   0,0
12466
12467  065256  000006              DF14:   .WORD   6               ;FORMAT FOR 2ND LEVEL ERROR
12468  065260     000      000             .BYTE   0,0             ;IN OPERATION INCOMPLETE ERROR
12469  065262  063221                      .WORD   DH101
12470  065264     007      000             .BYTE   7,0
12471  065266  063310                      .WORD   DH102
12472  065270     002      000             .BYTE   2,0
12473  065272  063613                      .WORD   DH601
12474  065274     000      000             .BYTE   0,0
12475  065276  064040                      .WORD   DH606
12476  065300     003      000             .BYTE   3,0
12477  065302  064424                      .WORD   DH502
12478  065304     000      000             .BYTE   0,0
12479
12480  065306  000011              DF15:   .WORD   11
12481  065310     000      000             .BYTE   0,0
12482  065312  063221                      .WORD   DH101
12483  065314     007      000             .BYTE   7,0
12484  065316  063310                      .WORD   DH102
12485  065320     002      000             .BYTE   2,0
12486  065322  063101                      .WORD   DH200
12487  065324     000      000             .BYTE   0,0
12488  065326  063413                      .WORD   DH201
12489  065330     007      000             .BYTE   7,0
12490  065332  063502                      .WORD   DH202
12491  065334     002      000             .BYTE   2,0
12492  065336  064465                      .WORD   DH503
12493  065340     000      000             .BYTE   0,0
```

```
12494  065342  063147                      .WORD   DH501
12495  065344     000      000             .BYTE   0,0
12496  065346  063517                      .WORD   DH203
12497  065350     010      000             .BYTE   10,0
12498
12499  065352  000011              DF16:   .WORD   11
12500  065354     000      000             .BYTE   0,0
12501  065356  063221                      .WORD   DH101
12502  065360     007      000             .BYTE   7,0
12503  065362  063310                      .WORD   DH102
12504  065364     002      000             .BYTE   2,0
12505  065366  063101                      .WORD   DH200
12506  065370     000      000             .BYTE   0,0
12507  065372  063413                      .WORD   DH201
12508  065374     007      000             .BYTE   7,0
12509  065376  063502                      .WORD   DH202
12510  065400     002      000             .BYTE   2,0
12511  065402  064424                      .WORD   DH502
12512  065404     000      000             .BYTE   0,0
12513  065406  063147                      .WORD   DH501
12514  065410     000      000             .BYTE   0,0
12515  065412  063517                      .WORD   DH203
12516  065414     010      000             .BYTE   10,0
12517
12518  065416  000005              DF17:   .WORD   5
12519  065420     000      000             .BYTE   0,0
12520  065422  063221                      .WORD   DH101
12521  065424     007      000             .BYTE   7,0
12522  065426  063310                      .WORD   DH102
12523  065430     002      000             .BYTE   2,0
12524  065432  064352                      .WORD   DH711
12525  065434     000      000             .BYTE   0,0
12526  065436  064175                      .WORD   DH702
12527  065440     002      000             .BYTE   2,0
12528
12529  065442  000002              DF20:   .WORD   2
12530  065444     000      000             .BYTE   0,0
12531  065446  063351                      .WORD   DH104
12532  065450     002      000             .BYTE   2,0
12533
12534  065452  000002              DF21:   .WORD   2
12535  065454     000      000             .BYTE   0,0
12536  065456  064003                      .WORD   DH6051
12537  065460     003      000             .BYTE   3,0
12538
12539  065462  000006              DF22:   .WORD   6
12540  065464     000      000             .BYTE   0,0
12541  065466  063050                      .WORD   DH100
12542  065470     000      000             .BYTE   0,0
12543  065472  063221                      .WORD   DH101
12544  065474     007      000             .BYTE   7,0
12545  065476  063310                      .WORD   DH102
12546  065500     002      000             .BYTE   2,0
12547  065502  063371                      .WORD   DH106
12548  065504     000      000             .BYTE   0,0
12549  065506  063351                      .WORD   DH104
```

```
12550  065510     002     000                    .BYTE   2,0
12551
12552  065512  000002                  DF23:     .WORD   2
12553  065514     000     000                    .BYTE   0,0
12554  065516  064542                            .WORD   DH800
12555  065520     001     000                    .BYTE   1,0
12556
12557  065522  000001                  DF24:     .WORD   1
12558  065524     002     000                    .BYTE   2,0
12559
12560  065526  000000                  DF25:     .WORD   0
12561  065530     007     000                    .BYTE   7,0
12562
12563  065532  000002                  DF26:     .WORD   2
12564  065534     002     000                    .BYTE   2,0
12565  065536  063517                            .WORD   DH203
12566  065540     010     000                    .BYTE   10,0
12567
12568  065542  000005                  DF27:     .WORD   5
12569  065544     000     000                    .BYTE   0,0
12570  065546  063221                            .WORD   DH101
12571  065550     007     000                    .BYTE   7,0
12572  065552  063310                            .WORD   DH102
12573  065554     002     000                    .BYTE   2,0
12574  065556  064211                            .WORD   DH703
12575  065560     000     000                    .BYTE   0,0
12576  065562  064175                            .WORD   DH702
12577  065564     002     000                    .BYTE   2,0
12578
12579  065566  000005                  DF30:     .WORD   5
12580  065570     000     000                    .BYTE   0,0
12581  065572  063221                            .WORD   DH101
12582  065574     007     000                    .BYTE   7,0
12583  065576  063310                            .WORD   DH102
12584  065600     002     000                    .BYTE   2,0
12585  065602  064255                            .WORD   DH705
12586  065604     000     000                    .BYTE   0,0
12587  065606  064235                            .WORD   DH704
12588  065610     004     000                    .BYTE   4,0
12589
12590  065612  000005                  DF31:     .WORD   5
12591  065614     000     000                    .BYTE   0,0
12592  065616  063221                            .WORD   DH101
12593  065620     007     000                    .BYTE   7,0
12594  065622  063310                            .WORD   DH102
12595  065624     002     000                    .BYTE   2,0
12596  065626  064271                            .WORD   DH706
12597  065630     000     000                    .BYTE   0,0
12598  065632  064334                            .WORD   DH710
12599  065634     003     000                    .BYTE   3,0
12600
12601  065636                          RWBUF:                ;READ/WRITE DATA BUF <AT LEAST 512(DEC) WORDS>
12602
12603  065636  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
12604  065644  020040  020040  020040
12605  065652  025052  020052  047516
```

```
12606  065660  042524  025040  025052
12607  065666  005015     012
12608  065671     101  046114  042040          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12609  065676  044522  042526  020123
12610  065704  047524  041040  020105
12611  065712  042524  052123  042105
12612  065720  046440  051525  020124
12613  065726  040510  042526  035040
12614  065734  005015     012
12615  065737     061  020056  020101          .ASCII  /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12616  065744  030062  047440  020122
12617  065752  031062  051440  041505
12618  065760  047524  020122  047506
12619  065766  046522  052101  042524
12620  065774  020104  040503  052122
12621  066002  044522  043504  006505
12622  066010     012
12623  066011     062  020056  042510          .ASCII  /2. HEADS MANUALLY LOADED/<15><12>
12624  066016  042101  020123  040515
12625  066024  052516  046101  054514
12626  066032  046040  040517  042504
12627  066040  006504     012
12628  066043     063  020056  042504          .ASCII  /3. DESIRED PORT SELECTED/<15><12>
12629  066050  044523  042522  020104
12630  066056  047520  052122  051440
12631  066064  046105  041505  042524
12632  066072  006504     012
12633  066075     064  020056  051127          .ASCII  /4. WRITE LOCK DISABLED/<15><12>
12634  066102  052111  020105  047514
12635  066110  045503  042040  051511
12636  066116  041101  042514  006504
12637  066124     012
12638  066125     065  020056  051104          .ASCII  /5. DRIVE READY LIGHT ON/<15><12><12>
12639  066132  053111  020105  042522
12640  066140  042101  020131  044514
12641  066146  044107  020124  047117
12642  066154  005015     012
12643  066157     104  044522  042526          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12644  066164  020123  047516  020124
12645  066172  047524  041040  020105
12646  066200  042524  052123  042105
12647  066206  046440  051525  020124
12648  066214  040510  042526  041040
12649  066222  052117  006510     012
```

```
12650  066227     120  051117  051524          .ASCIZ  /PORTS DE-SELECTED./<15><12><12>
12651  066234  042040  026505  042523
12652  066242  042514  052103  042105
12653  066250  006456  005012     000
12654
12655  066255     015  044412  051516  HLPFIL: .ASCII  <15><12>/INSTRUCTIONS FOR USING RK06-RK07 HEAD ALIGNMENT AID :/<15><12>
12656  066262  051124  041525  044524
12657  066270  047117  020123  047506
12658  066276  020122  051525  047111
12659  066304  020107  045522  033060
12660  066312  051055  030113  020067
12661  066320  042510  042101  040440
12662  066326  044514  047107  042515
12663  066334  052116  040440  042111
12664  066342  035040  005015
12665  066346  025052  025052  025052          .ASCII  /**********************************************/<15><12>
12666  066354  025052  025052  025052
12667  066362  025052  025052  025052
12668  066370  025052  025052  025052
12669  066376  025052  025052  025052
12670  066404  025052  025052  025052
12671  066412  025052  025052  025052
12672  066420  025052  025052  005015
12673  066426  047515  047125  020124          .ASCII  /MOUNT AN RK06 ALIGNMENT CARTRIDGE ON THE DESIRED /<15><12>
12674  066434  047101  051040  030113
12675  066442  020066  046101  043511
12676  066450  046516  047105  020124
12677  066456  040503  052122  044522
12678  066464  043504  020105  047117
12679  066472  052040  042510  042040
12680  066500  051505  051111  042105
12681  066506  006440     012
12682  066511     104  044522  042526          .ASCII  /DRIVE, AND INSURE THAT THE DRIVE IS WRITE-LOCKED. /<15><12>
12683  066516  020054  047101  020104
12684  066524  047111  052523  042522
12685  066532  052040  040510  020124
12686  066540  044124  020105  051104
12687  066546  053111  020105  051511
12688  066554  053440  044522  042524
12689  066562  046055  041517  042513
12690  066570  027104  006440     012
12691  066575     103  047117  042516          .ASCII  /CONNECT THE ALIGNMENT INDICATOR TO THE DESIRED DRIVE,/<15><12>
12692  066602  052103  052040  042510
12693  066610  040440  044514  047107
12694  066616  042515  052116  044440
12695  066624  042116  041511  052101
12696  066632  051117  052040  020117
12697  066640  044124  020105  042504
12698  066646  044523  042522  020104
12699  066654  051104  053111  026105
12700  066662  005015
12701  066664  044526  020101  044124          .ASCII  /VIA THE HEAD ALIGNMENT CABLE ONLY, AND CYCLE UP THE/<15><12>
12702  066672  020105  042510  042101
12703  066700  040440  044514  047107
12704  066706  042515  052116  041440
12705  066714  041101  042514  047440
```

```
12706  066722  046116  026131  040440
12707  066730  042116  041440  041531
12708  066736  042514  052440  020120
12709  066744  044124  006505     012
12710  066751     104  044522  042526          .ASCII  /DRIVE. AFTER MOUNTING THE PACK ON THE DRIVE, /<15><12>
12711  066756  020056  043101  042524
12712  066764  020122  047515  047125
12713  066772  044524  043516  052040
12714  067000  042510  050040  041501
12715  067006  020113  047117  052040
12716  067014  042510  042040  044522
12717  067022  042526  020054  005015
12718  067030  044124  020105  050117          .ASCII  /THE OPERATOR SHOULD WAIT 30 MINUTES FOR THE DRIVE/<15><12>
12719  067036  051105  052101  051117
12720  067044  051440  047510  046125
12721  067052  020104  040527  052111
12722  067060  031440  020060  044515
12723  067066  052516  042524  020123
12724  067074  047506  020122  044124
12725  067102  020105  051104  053111
12726  067110  006505     012
12727  067113     124  046505  042520          .ASCII  /TEMPERATURE TO STABILIZE, BEFORE PROCEEDING WITH /<15><12>
12728  067120  040522  052524  042522
12729  067126  052040  020117  052123
12730  067134  041101  046111  055111
12731  067142  026105  041040  043105
12732  067150  051117  020105  051120
12733  067156  041517  042505  044504
12734  067164  043516  053440  052111
12735  067172  020110  005015
12736  067176  046101  043511  046516          .ASCII  /ALIGNMENT./<15><12><12>
12737  067204  047105  027124  005015
12738  067212     012
12739  067213     122  051505  047520          .ASCII  /RESPOND TO ALL REQUESTS FOR PARAMETERS, BY ENTERING /<15><12>
12740  067220  042116  052040  020117
12741  067226  046101  020114  042522
12742  067234  052521  051505  051524
12743  067242  043040  051117  050040
12744  067250  051101  046501  052105
12745  067256  051105  026123  041040
12746  067264  020131  047105  042524
12747  067272  044522  043516  006440
12748  067300     012
12749  067301     124  042510  042040          .ASCII  /THE DESIRED PARAMETER VALUE (NO <CR> NEEDED)./<15><12>
12750  067306  051505  051111  042105
12751  067314  050040  051101  046501
12752  067322  052105  051105  053040
12753  067330  046101  042525  024040
12754  067336  047516  036040  051103
12755  067344  020076  042516  042105
12756  067352  042105  027051  005015
12757  067360  044124  051105  020105          .ASCII  /THERE ARE TWO MODES OF OPERATION :  MANUAL MODE /<15><12>
12758  067366  051101  020105  053524
12759  067374  020117  047515  042504
12760  067402  020123  043117  047440
12761  067410  042520  040522  044524
```

```
12762  067416  047117  035040  020040
12763  067424  040515  052516  046101
12764  067432  046440  042117  020105
12765  067440  005015
12766  067442  046101  047514  051527          .ASCII  /ALLOWS SELECTION OF DRIVES AND HEADS BY TTY INPUT,/<15><12>
12767  067450  051440  046105  041505
12768  067456  044524  047117  047440
12769  067464  020106  051104  053111
12770  067472  051505  040440  042116
12771  067500  044040  040505  051504
12772  067506  041040  020131  052124
12773  067514  020131  047111  052520
12774  067522  026124  005015
12775  067526  047101  020104  052501          .ASCII  /AND AUTO MODE ALLOWS DRIVES AND HEADS TO BE SELECTED/<15><12>
12776  067534  047524  046440  042117
12777  067542  020105  046101  047514
12778  067550  051527  042040  044522
12779  067556  042526  020123  047101
12780  067564  020104  042510  042101
12781  067572  020123  047524  041040
12782  067600  020105  042523  042514
12783  067606  052103  042105  005015
12784  067614  054502  047440  043106          .ASCII  /BY OFF-ON OPERATION OF DRIVE PORT SELECT SWITCHES./<15><12>
12785  067622  047455  020116  050117
12786  067630  051105  052101  047511
12787  067636  020116  043117  042040
12788  067644  044522  042526  050040
12789  067652  051117  020124  042523
12790  067660  042514  052103  051440
12791  067666  044527  041524  042510
12792  067674  027123  005015
12793  067700  047111  042440  052111          .ASCII  /IN EITHER MODE, UP TO 5 MINUTES OF SEEK EXERCISES/<15><12>
12794  067706  042510  020122  047515
12795  067714  042504  020054  050125
12796  067722  052040  020117  020065
12797  067730  044515  052516  042524
12798  067736  020123  043117  051440
12799  067744  042505  020113  054105
12800  067752  051105  044503  042523
12801  067760  006523     012
12802  067763     115  054501  041040          .ASCII  /MAY BE REQUESTED FOR EACH DRIVE. /<15><12>
12803  067770  020105  042522  052521
12804  067776  051505  042524  020104
12805  070004  047506  020122  040505
12806  070012  044103  042040  044522
12807  070020  042526  020056  005015
12808  070026  046101  047523  044440          .ASCII  /ALSO IN EITHER MODE, A VERIFY OPTION ALLOWS HEAD/<15><12>
12809  070034  020116  044505  044124
12810  070042  051105  046440  042117
12811  070050  026105  040440  053040
12812  070056  051105  043111  020131
12813  070064  050117  044524  047117
12814  070072  040440  046114  053517
12815  070100  020123  042510  042101
12816  070106  005015
12817  070110  042523  042514  052103          .ASCII  /SELECTION WITHOUT THE UNLOADING AND LOADING OF /<15><12>
```

```
12818  070116  047511  020116  044527
12819  070124  044124  052517  020124
12820  070132  044124  020105  047125
12821  070140  047514  042101  047111
12822  070146  020107  047101  020104
12823  070154  047514  042101  047111
12824  070162  020107  043117  006440
12825  070170     012
12826  070171     124  042510  042040          .ASCII  /THE DRIVE BY THE PROGRAM, WHICH OTHERWISE OCCURS/<15><12>
12827  070176  044522  042526  041040
12828  070204  020131  044124  020105
12829  070212  051120  043517  040522
12830  070220  026115  053440  044510
12831  070226  044103  047440  044124
12832  070234  051105  044527  042523
12833  070242  047440  041503  051125
12834  070250  006523     012
12835  070253     124  020117  046101          .ASCII  /TO ALLOW MANIPULATION OF THE ALIGNMENT TOOL./<15><12>
12836  070260  047514  020127  040515
12837  070266  044516  052520  040514
12838  070274  044524  047117  047440
12839  070302  020106  044124  020105
12840  070310  046101  043511  046516
12841  070316  047105  020124  047524
12842  070324  046117  006456     012
12843  070331     124  020117  042522          .ASCII  /TO RESTART EITHER MODE, TYPE <^Z>./<15><12>
12844  070336  052123  051101  020124
12845  070344  044505  044124  051105
12846  070352  046440  042117  026105
12847  070360  052040  050131  020105
12848  070366  057074  037132  006456
12849  070374     012
12850  070375     124  020117  042522          .ASCII  /TO RESTART ALIGNMENT AID, TYPE <^R>./<15><12>
12851  070402  052123  051101  020124
12852  070410  046101  043511  046516
12853  070416  047105  020124  044501
12854  070424  026104  052040  050131
12855  070432  020105  057074  037122
12856  070440  006456     ^12
12857  070443     124  020117  042523          .ASCII  /TO SELECT NEW DRIVES IN MANUAL MODE, TYPE <^C>./<15><12><12>
12858  070450  042514  052103  047040
12859  070456  053505  042040  044522
12860  070464  042526  020123  047111
12861  070472  046440  047101  040525
12862  070500  020114  047515  042504
12863  070506  020054  054524  042520
12864  070514  036040  041536  027076
12865  070522  005015     012
12866  070525     106  051117  044040          .ASCII  /FOR HEAD ALIGNMENT PROCEDURE, REFER TO FIELD /<15><12>
12867  070532  040505  020104  046101
12868  070540  043511  046516  047105
12869  070546  020124  051120  041517
12870  070554  042105  051125  026105
12871  070562  051040  043105  051105
12872  070570  052040  020117  044506
12873  070576  046105  020104  005015
```

```
 12874  070604  042524  052123  041040          .ASCIZ  /TEST BOX (RK06-07TA, RK06-07TB) OPERATOR'S MANUAL./<15><12>
 12875  070612  054117  024040  045522
 12876  070620  033060  030055  052067
 12877  070626  026101  051040  030113
 12878  070634  026466  033460  041124
 12879  070642  020051  050117  051105
 12880  070650  052101  051117  051447
 12881  070656  046440  047101  040525
 12882  070664  027114  005015     000
 12883
 12884
 12885
 12886          000001                  .END
```

```
ABASE = 177440          2245#   2337    2378    4795
ABORT = 000200          4235#   7443    9460    9465    9548    9573    9661    9671    9676    9681    9686    9694    9700
                        9710    9748
ACDW1 = 000000          2337
ACDW2 = 000000          2337
ACLO  = 000010          3291#   9510
ACPUOP= 000000          2337    2352
ADDW0 = 000000          2337
ADDW1 = 000000          2337
ADDW10= 000000          2337
ADDW11= 000000          2337
ADDW12= 000000          2337
ADDW13= 000000          2337
ADDW14= 000000          2337
ADDW15= 000000          2337
ADDW2 = 000000          2337
ADDW3 = 000000          2337
ADDW4 = 000000          2337
ADDW5 = 000000          2337
ADDW6 = 000000          2337
ADDW7 = 000000          2337
ADDW8 = 000000          2337
ADDW9 = 000000          2337
ADEVCT= 000000          2337    2343
ADEVM = 000377          2246#   2337    2379
AENV  = 000000          2337    2348
AENVM = 000000          2337    2349
AFATAL= 000000          2337    2340
ALDRVS  007656          4320#   4862
ALLDRV  015060          4801    4871    4963#
ALLTRK  045752          7444    9549    9684    9691#
ALNCYL= 000365          4228#   7602
ALNPAK  007533          4304#   7517
ALNSEK  033460          6527    6793    7596#
AMADR1= 000000          2337    2365
AMADR2= 000000          2337    2369
AMADR3= 000000          2337    2372
AMADR4= 000000          2337    2375
AMAMS1= 000000          2337    2359
AMAMS2= 000000          2337    2367
AMAMS3= 000000          2337    2370
AMAMS4= 000000          2337    2373
AMSGAD= 000000          2337    2345
AMSGLG= 000000          2337    2346
AMSGTY= 000000          2337    2339
AMTYP1= 000000          2337    2360
AMTYP2= 000000          2337    2368
AMTYP3= 000000          2337    2371
AMTYP4= 000000          2337    2374
ANYDER= 100000          4241#   7509    9171    9736
APASS = 000000          2337    2342
APRIOR= 000000          2337
APTCSU= 000040         11073   11733#
APTENV= 000001          6277   11066   11512   11689   11731#
APTPAS  024042          6272    6277#
APTSIZ= 000200          4739    6326   11730#
```

```
APTSPO= 000100  11068  11691  11732#
ASKDRV  024756   6414#  6456   6460   6475   6507   6577
ASKHED  025402   6493#  6521   6533   6540
ASKMDE  015760   5131#  5137   5138   5143   5158   5163   5168   5193   5207   5218   5227   5243   5262
                 5271   5297   5320   5392   7950
ASKMOD  024664   6392#  6406
ASKTMD  015214   5013#  5016   5017   5022   5046   5054   5060   5076   5087   5094   5121
ASKTST  015210   4929   5012#
ASKTYP  024560   6361   6367   6370#
ASTART  024074   2208   6286#
ASWREG= 000000   2337   2350
AS2SP2  060346   9218   11901#
ATESTN= 000000   2337   2341
AUNIT = 000000   2337   2344
AUSWR = 000000   2337   2351
AUTAL   026226   6604   6611   6618#  6739   6760
AUTALN  012711   4625#  6623
AUTEX   027460   6607   6818#  6839   6880
AUTEXR  013025   4640#  6821
AUTO    026104   6404   6590#  6639   6807   6810   6813   6829   6903   6918
AUTODR  013135   4653#  6884*
AUTOEX  013116   4650#  6885
AUTOFG  003150   3678#  4743*  6330*  6412*  6591*  7457   7473   7483   7506
AUTVRF  012747   4631#  6621
AVECT1= 120210   2244#  2337   2376   4794
AVECT2= 000000   2337   2377
A.ABNL  003046   3572#  5600*  5603*  6651*  6720*  6842*  6875*  7369*  7450*  7454*  7545*  7549*  8490*
                 8786*  8803*  8812*  9436*  9692*  9705*  9707*  10634
A.CONT  003050   3573#  10644
A.NORM  003044   3571#  6166*  6227*  7370*  9437*  9693*  10638
BADDRV  007461   4296#  6374*  6375   7459*  7460*  7461   7475*  7476*  7477   7485*  7486*  7487   7514*
                 7515*  7516
BADSEC= 001000   4237#  8399   8622   9075   9083   9638   9741
BAD632  013141   4655#  7511
BAI   = 000020   3250#  10803
BAIA    025246   6454   6457#
BAIFLU  025240   6422   6424   6455#
BA16  = 000400   3226#
BA17  = 001000   3227#
BDSCHD  041332   8786   8803   8817#
BDSECT  011150   4455#  8835
BDSRCK  042404   8397   8620   9065#  9637
BGNRTY  046144   9360   9703   9709   9734#
BITO  = 000001   2136#  3223   3240   3266   3288   3436   3580   3672   4879   4968   7171   7989   8273
                 8290   8305   8344   8361   8415   8457   8562   8577   8608   8644   8688   9315   10918
BITOO = 000001   2126#  2136
BIT01 = 000002   2125#  2135
BIT02 = 000004   2124#  2134
BIT03 = 000010   2123#  2133
BIT04 = 000020   2122#  2132
BIT05 = 000040   2121#  2131
BIT06 = 000100   2120#  2130
BIT07 = 000200   2119#  2129
BIT08 = 000400   2118#  2128   8114
BIT09 = 001000   2117#  2127   11522  11653
BIT1  = 000002   2135#  3241   3268   3437   3581   3673   4229   5577   5628   8532   9319
```

```
BIT10 = 002000   2116# 3228  3257  3279  3314  3328  3341  3356  3370  3449  3590  4238  9350
                 11500
BIT11 = 004000   2115# 3229  3258  3280  3299  3315  3329  3342  3357  3371  3450  4239  11660
BIT12 = 010000   2114# 3231  3259  3281  3316  3330  3343  3344  3358  3372  3451  4240  9230
                 9258
BIT13 = 020000   2113# 3232  3260  3282  3300  3317  3331  3345  3359  3373  3452  7349  7351
                 8296  8299  8370  8373  8569  8572  8679  8682  11507
BIT14 = 040000   2112# 3233  3261  3283  3301  3318  3332  3346  3360  3374  3453  3591  11635
BIT15 = 100000   2111# 3234  3235  3262  3284  3302  3319  3379  3454  3592  4241  5204  7880
                 9354
BIT2  = 000004   2134# 3242  3269  3270  3290  3438  3582  3674  4230  9323
BIT3  = 000010   2133# 3248  3249  3271  3291  3439  3583  3675  4231  9330
BIT4  = 000020   2132# 3243  3250  3272  3292  3293  3308  3350  3364  3440  3584  3676  4232
                 5487  9334
BIT5  = 000040   2131# 3251  3252  3273  3294  3309  3323  3336  3351  3365  3441  3585  4233
                 6953  9195  9338  10715
BIT6  = 000100   2130# 3224  3253  3274  3295  3310  3324  3337  3352  3366  3442  3587  4234
                 5053  5206  5226  6248  6562  6916  6958  6991  8442  8671
BIT7  = 000200   2129# 3225  3254  3275  3296  3297  3311  3325  3338  3353  3367  3444  4235
                 4792  4965  5463  5466  5617  5622  7463  9838  9841
BIT8  = 000400   2128# 3226  3255  3276  3277  3298  3312  3326  3339  3354  3368  3445  3588
                 4236  9342
BIT9  = 001000   2127# 3227  3256  3278  3313  3327  3340  3355  3369  3446  3589  4237  5472
                 7137  9346  11612
BKSLSH  013242   4668# 7746  7754
BLDEXH  047004   9589  9614  9885#
BPTVEC= 000014   2143#
BSE   = 000200   3275# 8817  9629  10068 10234 10470
BSERR = 000002   4229# 6016  6169  8997  9631  9641  9734
BSFACT  004224   3696# 6087  6090  8785  8833  9066
BSSOFT  003224   3695# 6079  8797  8866  9071  9073
BUFFO   005264   3705# 4838  4869  4872  4933  4937  4939  4944  4955  4959  5020  5023  5026
                 5029  5064  5066  5097  5100  5102* 5107  5125  5141  5144  5147  5150  5153
                 5247  5249  5281  5298  5324  5331  5341  5343  5346  5351  5360  5398  5401
                 7082  7086  7749  7763* 7768* 7769  7773* 7890  7892  7912  7915  7917* 7922
                 7938
B.BA16= 000001   3240# 10797
B.BA17= 000002   3241# 10797
B.CDT = 000004   3242# 5439  6155  6209  6386  7447  10451 10797
B.CFMT= 000020   3243# 5474  7504  8084  8775  9843  10451 10797 10881
CART    011101   4447# 8094
CCLR  = 100000   3235# 10404
CDT   = 002000   3228# 7432  10210 10760 10817 10868 10891
CERR  = 100000   3234# 9598  9623  10106 10165 10224 10406 10457 10772 10828 10916
CFMT  = 010000   3231# 9864  9897  10210 10760 10817 10868 10891
CHGTST  015432   5028  5057# 5061  5068
CHKDRV  033220   6459  6759  6879  7543#
CHKITR  033706   5533  5722  5799  5887  5988  6071  7662#
CHKLST  015136   4844  4918  4961  4978#
CHKPRM  035006   5287  5366  7971#
CKMEM1  040022   5766  8591#
CKMEM2  040034   5851  8595#
CKMO    040042   8594  8597#
CLEAR = 000105   3194#
CLR   = 000040   3251#
CLRPRM  032356   7374  7393#
```

```
CMDTO = 000100  3442#  9483   9798   9967
CMPBUF  036444  5941   8331#
CMSTRT  013350  4693   4699#  9725
CNTRDY  007613  4313#  8503
CNTRLC  013205  4662#  6957   7717
CNTRLG  013231  4666#  6971   7735
CNTRLR  013217  4664#  6967
CNTRLU  013224  4665#  7729
CNTRLZ  013212  4663#  6963   7723
COE   = 001000  3278#  9555   10068  10234  10470
COMMA   013244  4669#  4884
COMSTR  005240  3699#  9136   9151   9753
CONCLR= 000176  3212#  10899
CONERR  043442  3573   9311#
CR    = 000015  2051#  11112  11122
CRLF  = 000200  2052#  11083  11122
CRMPHO  005262  3704#  8405   8634   9146   9148*
CRMPLO  005260  3703#  8406   8635   9147   9149*
CRTSER  035432  5486   8083#
CR2LF   013236  4667#  9217   9229   9253
CS      005702  3804#  5487   5577   5628   9195
CTLOUT  037352  8474#  8725   9111
CTO   = 004000  3229#  9518   10466
CYL     003202  3693#  6098*  6120   6165
CYLLST  003204  3694#  6133   6141   6176
CYLNDR  005504  3712#  5546*  5549*  5571   5573   5665   5667   6135*  6136   6555   6890   8142*  8143*
                8144   8146*
C.INIT  052562  6158   8730   9121   10677#
C.RTRN  054434  10781  10809  10837  10844  10847  10854  10858  10875  10897  10902  10912  10921  10934
                10938  10943#
C.SPEC  053454  10717  10813#
DCK   = 100000  3284#  5691   8817   9643   10068  10234  10470
DCKERR= 000020  4232#  9645   9734
DCLO  = 000020  3293#
DDISP = 177570  2058#  2275   4727   6314
DDT   = 000400  3298#
DERCNT  003121  3649#  9582   9607   9652   9663
DESL  = 000010  3248#
DFLTST  005644  3781#  3790   5051   5070   5073   5094   5119
DFQUES  010534  4404#  5240
DFSTRT  013302  2202   4687#
DFTEST  010067  4347#  5057
DF01    064702  2402   2408   2414   2420   2462   2570   2840   2876   12338#
DF02    064732  2426   2444   2450   2456   2468   2474   2480   2486   2492   2498   2504   2510   2516
                2522   2528   2534   2588   2594   2600   2708   2714   2720   2726   2732   2738   2744
                2750   2756   2762   2804   2822   2846   12352#
DF03    064766  2432   2438   2810   2816   12367#
DF04    065026  2564   12384#
DF05    065042  2540   2546   12391#
DF07    065076  2552   12406#
DF10    065132  2558   12421#
DF11    065156  2858   12432#
DF12    065176  2606   2612   2618   2624   2630   2636   2642   2648   12441#
DF13    065226  2654   2666   12454#
DF14    065256  2660   12467#
DF15    065306  2672   12480#
```

```
DF16    065352   2678  12499#
DF17    065416   2696  12518#
DF20    065442   2864  12529#
DF21    065452   2774  12534#
DF22    065462   2870  12539#
DF23    065512   2780   2786   2792  12552#
DF24    065522   2798  12557#
DF25    065526   2702   8333*  8387*  8601*  8627*  12560#
DF26    065532   2852  12563#
DF27    065542   2828  12568#
DF30    065566   2882   7155*  12579#
DF31    065612   2888   7198*  7199*  12590#
DH100   063050   2400   2406   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   2586   2592   2598   2604   2610   2616   2622   2628   2634   2640
                 2646   2652   2658   2664   2670   2676   2694   2706   2712   2718   2724   2730   2736
                 2742   2748   2754   2760   2802   2808   2814   2820   2826   2838   2844   2874   2880
                 2886  12158#  12541
DH101   063221   9234  12179#  12341  12354  12369  12386  12393  12408  12423  12443  12456  12469  12482
                12501  12520  12543  12570  12581  12592
DH102   063310   9245  12189#  12343  12356  12371  12388  12395  12410  12425  12445  12458  12471  12484
                12503  12522  12545  12572  12583  12594
DH103   063326   2862  12192#
DH104   063351  12196#  12531  12549
DH105   063066   9232  12161#
DH106   063371  12199#  12547
DH200   063101  12163#  12345  12358  12373  12447  12486  12505
DH201   063413  12203#  12347  12360  12375  12449  12488  12507
DH202   063502  12213#  12362  12379  12490  12509
DH203   063517  12216#  12364  12381  12496  12515  12565
DH204   064366  12297#  12451
DH500   063116  12166#  12349
DH501   063147  12171#  12377  12494  12513
DH502   064424  12302#  12464  12477  12511
DH503   064465  12308#  12492
DH601   063613  12227#  12397  12460  12473
DH602   063635  12231#  12401
DH604   063673  12237#  12412  12427  12434
DH6041  063721   8836  12241#  12414  12429  12436
DH6042  063750   2796  12245#
DH605   063766   2772  12248#  12416
DH6051  064003  12251#  12418  12536
DH606   064040  12256#  12399  12403  12462  12475
DH607   064066  12260#
DH701   064106   8332*  8386*  8600*  8626*  12263#  12438
DH702   064175   2850  12273#  12526  12576
DH703   064211  12276#  12574
DH704   064235   7156*  12280#  12587
DH705   064255  12283#  12585
DH706   064271  12286#  12596
DH707   064304   7198  12288#
DH710   064334  12292#  12598
DH711   064352  12295#  12524
DH800   064542  12316#  12554
DI    = 040000   3233#  10254  10926
DISPLA  001142   2275#   4727*  4735*  6314*  6322*  11499*  11673*
```

```
DISPRE  000174          2199#  4735   6322
DLT   = 100000          3262#  9575   10062  10192  10228  10468
DLTFLG  003141          3665#  6010*  6014   6025
DMD   = 000040          3309#
DONE    003123          3651#  8726*  8733   9112*  9124   9170*  9312*  9430*  9767*  9770*
DRA   = 000001          3288#  10057  10840
DRDY  = 000200          3297#
DRIEXR  012344          4580#  6549*
DRISEL  013002          4636#  6758
DRIV    011074          4446#  8052
DRIVE   005500          3710#  4981*  5409*  5416*  5417   5420   5480   6108   6148   6164   6167   6182   6426*
                        6427   6547   5649*  6654   6682   6691   6693*  6721*  6726*  6753   6755   6764   6778
                        6840*  6843   6844   6882   6910*  6911   6931   7377   7378   7425   7427   7459   7475
                        7485   7514   7543   7552   7560   7577   12321
DRNAFG  003136          3662#  9113*  9325
DRNAVL= 010000          4240#
DRNRDY  011561          4512#  7562*
DROPDR  011010          4436#  9201
DROT  = 000040          3294#  9565
DRPAR = 000010          3271#  9505
DRPDRV= 002000          3449#  10702
DRVCAL  042510          5461   5595   5598   5602   5605   5608   5676   6019   6124   6172   6212   6228   6501
                        6506   6511   6556   6564   6571   6576   6581   6690   6738   6748   6752   6769   6771
                        6775   6785   6874   6891   6897   6902   6909   7376   7452   7492   7494   7508   7547
                        7571   7584   7586   7598   7616   7618   7630   7646   8051   8093   8179   8493   8534
                        8537   8788   8805   8993   9108#  9706   9752   9762   9797   9834   9836
DRVDSC= 000040          3441#  9673   10142  10308  10370
DRVERS  003130          3656#  4751*  5414*  9194*  9198   9200*  9714*
DRVHRD= 000020          3440#  9668   10149  10344  10475
DRVLST  005610          3745#  4880   4899   4915   4949   4957*  4969*  4972*  4979   4991*  5421   6106   6146
                        6180
DRVLUP  026304          6630#  6701   6716   6814
DRVMSK= 000007          3247#  10051  10092  10243  10688
DRVNED  011713          4529#  7554*
DRVNO   011070          4445#  5480*  5481*
DRVPDT= 000004          3438#  10250  10318
DRVPOS= 000002          3437#  10250  10277  10739
DRVSEL  013007          4637#  6757*
DRVSEQ  007445          4293#  4876
DRVSER  035312          5485   6461   8048#
DRVSZD= 010000          3451#  9496   10080  10842
DRVTST  014374          4690   4842   4858#  4864   4865   4866   4874   4922   4923   5015   5043   5059   5086
                        5136   5190   5215   5242   5270   5319   5391   7947   8482
DRVUSE= 000001          3436#  10702
DRY   = 000200          3296#
DR.CLR= 000005          3219#  10130  10293  10335  10359  10473
DR.SEL= 000001          3218#  10200  10279  10594  10601  10608  10615  10819
DSC   = 040000          3301#
DSWR  = 177570          2057#  2274   4726   6313
DTBAII= 100000          3454#  6001   6127   10702  10801
DTE   = 010000          3281#  5691   8817   9570   10470
DTYE  = 000040          3273#  7445   9533   10470
DTYLST  005620          3755#  4986*  5425   6153   6207   6382*  6386*  6428   6655   6845
DTYMSG  012513          4599#  6371
DT100   064566          2401   2407   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	2587	2593	2599	2605	2611	2617	2623	2629	2635	2641
		2647	2653	2659	2665	2671	2677	2695	2707	2713	2719	2725	2731	2737
		2743	2749	2755	2761	2803	2809	2815	2821	2827	2839	2845	2869	2875
		2881	2887	12321#										
DT102	064604	12324#												
DT103	064676	2863	12336#											
DT201	064610	12325#												
DT202	064626	2851	12328#											
DT203	064632	12329#												
DT601	064652	2773	2779	2785	2791	2797	2857	12332#						
DT602	064660	2701	12333#											
DUACES	010347	4380#	4762											
DULACS	003135	3661#	4688*	4697*	4760	4927	7664							
DUNPAS	023714	5419	6247#											
DZR6N	007036	4243#	4753	6333										
ECCNC =	000040	4233#	9648											
ECCW =	020000	3317#												
ECH =	000100	3274#	9646	10068	10234	10470								
ECOBAD	030556	5047	5194	5219	6405	6455	6484	6520	6583	6612	6647	6838	6904	7005#
		7627												
EMTVEC=	000030	2146#	4710*	4711*	6297*	6298*								
EM1	060353	2399	11902#											
EM10	060522	2441	11923#											
EM100	062325	2789	12096#											
EM101	062350	2795	12100#											
EM102	062426	2801	12108#											
EM103	062472	2807	12114#											
EM104	062520	2813	12118#											
EM105	062542	2819	12121#											
EM106	062573	2825	12126#											
EM107	062626	2831	12131#											
EM11	060531	2447	11925#											
EM110	062671	2837	12137#											
EM111	062724	2843	12142#											
EM112	062745	2849	12145#											
EM113	062771	2873	12149#											
EM114	063017	2879	12153#											
EM115	063034	2885	12156#											
EM12	060544	2453	11927#											
EM13	060556	2459	11929#											
EM14	060567	2465	11931#											
EM15	060607	2471	11934#											
EM16	060623	2477	11937#											
EM17	060634	2483	11939#											
EM2	060372	2405	11905#											
EM20	060651	2489	11942#											
EM21	060664	2495	11944#											
EM22	060700	2501	11946#											
EM23	060713	2507	11948#											
EM24	060730	2513	11951#											
EM25	060746	2519	11954#											
EM26	060765	2525	11957#											
EM27	060777	2531	11959#											
EM3	060410	2411	2890	11908#										
EM30	061015	2537	2657	11962#										
EM31	061036	2543	2663	11965#										

```
EM32    061055          2549   11968#
EM33    061074          2555   11971#
EM34    061110          2561    2861   11973#
EM35    061130          2567   11976#
EM36    061162          2573   11981#
EM37    061211          2579   11985#
EM4     060426          2417   11911#
EM40    061252          2585   11991#
EM41    061316          2591   11997#
EM42    061346          2597   12001#
EM43    061376          2603   12005#
EM5     060445          2423   11914#
EM52    061427          2645   12010#
EM53    061444          2651   12013#
EM56    061467          2669    2675   12017#
EM6     060464          2429   11917#
EM60    061506          2681   12020#
EM61    061540          2687   12025#
EM62    061573          2693   12030#
EM63    061607          2609    2705   12033#
EM64    061645          2615    2711   12039#
EM65    061673          2621    2717   12043#
EM66    061715          2627    2723   12047#
EM67    061741          2633    2729   12051#
EM7     060503          2435   11920#
EM70    062000          2639    2735   12057#
EM71    062045          2741   12064#
EM72    062105          2747   12070#
EM73    062140          2753   12075#
EM74    062160          2759   12078#
EM75    062172          2765   12080#
EM76    062222          2771   12084#
EM77    062304          2777    2783   12093#
ENBCSR  031302          4749    4861    5132    7165#   7211
ENTDRV  011521          4505#   6414
ENTHED  012557          4607#   6493
ENTLCI  010041          4343#   5013
ENTMOD  012212          4563#   6465    6595
ENTPAS  010742          4429#   5389
EQUALS  013270          4678#   7849
ERRCNT  003126          3654#   4689*   4841*   6357*   7643*   9173*   9175    9178    9180*   9313*   9431*   9715*   9723*
                        9743    9746
ERRCOM  005476          3709#   9368*   9369*   12321
ERRFRE  042700          3571    6227    6229    7370    9170#   9693
ERRHDL  044156          3572    5603    5693    6720    6875    6926    7369    7454    7549    8490    8812    8819    9429#
                        9692    9707
ERRLMT  003127          3655#   9743
ERRVEC= 000004          2139#   4724    4725*   4736*   4745*   4746*   6311    6312*   6323*   7150*   7151*   7166    7167
                        7168*   7177*   7178*   7188    7189    7192*   7193*   7208*   7209*   7215*   8729*   8744*   11640
                       11641*  11643*  11646*  11754   11755   11759*  11764*  11776*  11781*  11790*  11800*  11801*
ER2ENT  044176          9433#   9495    9603    9628    9712
EXPLAN  010173          4360#   5130
E.CCLR= 000001          3580#  10409
E.CERR= 001000          3589#  10483
E.CLAT= 000020          3584#  10135   10299   10340   10364
E.CMTO= 040000          3591#
```

```
E.CONT  003052   3574#  9315   9319   9323   9330   9334   9338   9342   9346   9350   9354   9359*  10034*
                10072*  10135*  10145*  10216*  10238*  10261*  10267*  10299*  10340*  10364*  10409*  10461*  10483*
                10622*  10856*  10910*  10918*  10931*
E.DLT = 000400   3588#  10216
E.DPAR= 002000   3590#  10145  10622
E.ILLD= 000100   3587#  10910
E.MDS = 100000   3592#  10034  10461  10856
E.NOAT= 000002   3581#  10261
E.SCLR= 000040   3585#  10931
E.UATT= 000004   3582#  10267
E.UDAT= 000010   3583#  10072  10238
E.UNLD= 020000   3452#  10153  10273  10348  10358
FACTBS  011123   4451#  8834
FC      005660   3795#  5543   5581   5673   5994   6077   6344*  8697   8700
FINADR  036040   8216#  8392   8615
FINCYL  005566   3732#  8246*  8394   8419   8425   8617   8648   8654
FINMEM  041530   8878#  8928
FINSEC  005571   3734#  8234*  8396   8423   8427   8619   8652   8656
FINTRK  005570   3733#  8244*  8395   8421   8426   8618   8650   8655
FMTE  = 000020   3272#  9537   10470
FNDXDP  031730   5728   5805   5893   7272#
FORHLP  011370   4489#  6362
FORMAT  003125   3653#  5468*  5474*  5679   5998   6123   7379   8085   8226   8780   8800   8905   9844
FS      005506   3713#  5469*  5475*  5541*  5544   5574   5582   5668   5677*  5681*  8698
FT      005664   3797#  5164   5580   5995   8699
GETPRM  030756   4811   4816   4825   7068#  7108
GETXDP  032010   4786   5771   5859   5947   7299#  8485
GIVEID  024452   6346#  6398   6420   6472   6502   6572   6602   6646   6826   6898
GNS   = ****** U  2198  11887  11888  11889  11890  11891  11894  11895  11896  11897
GO    = 000001   3223#  7430   7432
GTPKAD  046660   9654   9665   9854#
GTPRMS  042102   8973#  9005
GTSWRG  031066   4780   7104#  7736   8508
GTO     042114   8969   8972   8976#
HCRC  = 000400   3276#
HDALI   005532   3723#
HDR.AD  003070   3614#  10186  10190*  10880*
HDR.CT  003072   3615#  10194*  10883*  10886*
HEADNO  012643   4617#  6531*  6797*
HEDPOS  012447   4592#  6528   6794
HEDSEL  012636   4616#  6532   6798
HIERAD= 177742   2190#  7196
HIGOCT  005472   3707#  5285*  5365*  7820*  7823*  7976   7991   7996   8003
HLPFIL  066255   6368   12655#
HLPOVL  003145   3670#  4752*  6359
HLTPRG  046126   4904   7158   7207   8504   8796   9721#  9749
HLTRQD  007643   4318#  8502
HOLD2   020002   5434*  5446*  5495#  6434*  6447*  6662*  6673*  6852*  6863*  8146
HOLD3   020004   5435*  5447*  5462   5496#  6435*  6445*  6663*  6674*  6853*  6864*  9837
HOLD4   020006   5436*  5448*  5465   5497#  6436*  6446*  6664*  6675*  6854*  6865*  9840
HT    = 000011   2049#  11081  11122
HVRC  = 000400   3277#  5691   8817   9604   10068  10234  10470
HVRCER= 000004   4230#  9606   9734
H.HEAD  003075   3618#
IDAE  = 002000   3279#  9560
IDENT   011324   4483#  6358
```

```
IE     = 000100   3224#  6734   7426   7467   7565   9594   9619   10414  10478  10766  10807  10873  10940
ILC    = 000001   3266#  9521   10470
ILF    = 000004   3269#  9529
INCRMA  041020    5769   5847   5853   5945   8751#
INCRSC  041614    8890   8904#  8932
INITMM  031120    7124#  7237   7326
INITSS  032250    4917   5459   5478   5536   5725   5802   5890   5991   6074   6119   6457   6592   6819
                  7369#  7424   7448   8049   8083   8774
INPUTP  015754    4930   5031   5130#  8498
INTCHR  005522    3719#  5039   5041   5044   5186   5188   5191   5211   5213   5216   5415*  5490*  6380
                  6383   6394   6396   6416   6418   6467   6469   6495   6497   6558   6567   6597   6599
                  6630   6633   5640   6803   6805   6808   6811   6822   6824   6827   6830   6834   6892
                  6894   6899   '954*  6973   6990*  7005   7623   7625   8476   8480   8487*  8496   8500
                  8506
INTG    026370    6651#  6696
INTMSK  003101    3623#  10109  10133  10154  10179  10265  10297  10338  10349  10362  10633* 10637* 10641*
                  10690* 10776
INTR   = 000300   3214#
IOTVEC= 000020    2144#  4708*  4709*  6295*  6296*
IR     = 000100   3253#
I.AERR  051144    10285  10291  10309  10318#
I.ATTN  050474    10045  10089  10222#
I.CCLR  051530    10123  10173  10404#  10476  10836  10901
I.CSTS  052162     6220  10116  10551#
I.CST1  052204     7440  10120  10172  10555#  10825
I.DRV   003102     3627#  6184  10690  10691
I.ERR   047716    10117  10121#
I.ERRA  047676    10116#  10184
I.ERRC  047704    10054  10107  10119#  10252
I.HDAL  050142    10100  10163#
I.INTR  047262     4766   6341  10024#
I.ISRL  003074     3617#  10038  10041*  10044*  10415*  10712*  10866*  10939*
I.ISSU  051612    10131  10201  10280  10294  10336  10360  10446#  10595  10602  10609  10616
I.IOO   047542    10042  10059  10086#
I.RTRN  051512    10036  10074  10084  10112  10124  10129  10132  10138  10147  10156  10159  10174  10176
                  10202  10205  10214  10218  10240  10256  10263  10269  10281  10295  10302  10314  10334
                  10337  10342  10351  10354  10361  10366  10375#
I.STAT  052266    10128  10333  10592#  10846
I.STOR  052034    10456  10506#
I.UNLD  051412    10275  10358#
KBDHDL  030376     4763   6338   6946#
KILLDR  000170     2204#  7304   8704
KIPAR0= 172340     2177#  7126  11740  11768
KIPAR1= 172342     2178#
KIPAR2= 172344     2179#
KIPAR3= 172346     2180#
KIPAR4= 172350     2181#
KIPAR5= 172352     2182#
KIPAR6= 172354     2183#  7334*  7352*  8289*  8298*  8360*  8368   8372*  8561*  8571*  8607*  8629   8681*
KIPAR7= 172356     2184#  11784
KIPDR0= 172300     2166#
KIPDR1= 172302     2167#
KIPDR2= 172304     2168#
KIPDR3= 172306     2169#
KIPDR4= 172310     2170#
KIPDR5= 172312     2171#
```

```
KIPDR6= 172314          2172#
KIPDR7= 172316          2173#
KTERHD  031206          4768    7146#
LASTWC  005572          3735#   8217    8391*   8614*
LC      005662          3796#   5430*   5442*   5546    5547    6431*   6442*   6453*   6658*   6669*   6689*   6848*   6859*
LCM6    017776          5428*   5440*   5493#   6429*   6440*   6656*   6667*   6846*   6857*   8700    8702
LCVEC   003162          3684#
LDMEM1  037656          5759    8549#
LDMEM2  037670          5763    5839    5843    5938    8553#
LDMO    037676          8552    8555#
LEV2ER= 000400          4236#   9433    9488    9585    9591    9610    9616    9708    9768
LF    = 000012          2050#   11116   11122
LKS     003152          3680#
LODBUF  036216          5934    8264#
LODFLS  015476          4913    5065    5070#
LODFPT  016510          5077    5248    5253#
LODP14  036002          5899    8193#
LODSEC  035674          8155#   8177
LOERAD= 177740          2189#   7195
LOWOCT  005470          3706#   5284*   5364*   7819*   7824    7981    7983    7986    7989    7994    7999    8002
LS      005510          3714#   5470*   5476*   5542*   5566    5568    5661    5663    5678*   5682*
LSTCYL  020000          5431*   5443*   5494#   5547    5557    6083    6437*   6448*   6453    6659*   6670*   6689    6849*
                        6860*   8089    8144    8777
LSTMEM  007101          4250#   7051
LSTTRK= 000002          4227#   7501    8090
LSTTST  015314          5025    5034#
LT      005666          3798#   5164
MA      005600          3738#   5745    5746    5752    5753    5824    5825    5831    5832    5913    5914    5920    5921
                        8026    8028    8703*   8706*   8707*   8708*   8709*   8761*   8762*   8763*   8970    8971
MAHILM  005574          3736#   7023*   7024*   7038*   7039*   7052    7272    7276    7278    8024    8025
MANAL   025376          6477    6490#
MANALN  012352          4581#   6491
MANEX   025650          6480    6545#
MANEXR  012265          4571#   6550    6584
MANUAL  024746          6401    6411#
MANVR   025632          6483    6538#
MANVRF  012412          4587#   6539
MAPH00= 170202          2187#   9148
MAPL00= 170200          2186#   7254    9149
MAWRDC  010405          4386#   8006
MAXHI1= 000000          4218#
MAXHI2= 000000          4220#
MAXHI3= 000000          4222#
MAXHI4= 000001          4224#
MAXLO1= 062031          4217#
MAXLO2= 017500          4219#
MAXLO3= 112160          4221#
MAXLO4= 022370          4223#
MCLK  = 000400          3312#
MDFLAG  003116          3646#   4687*   4696*   4788    4911    5075    5258    5539    6349*   7666    8478
MDFY15  010667          4421#   7883
MDS   = 001000          3256#   7436    10032   10459   10834
MEMABT  003144          3668#   7205
MEMCSR= 172100          2188#   7169    7175    7216    7224
MEMSYS= 177744          2191#   7197
MEMVEC= 000114          3686#   4747*   4748*
```

```
MERD   = 001000          3313#
MESMSK= 000017          3306#
MEWD   = 002000          3314#
MIDXFR  041660          8927#   9000
MIND   = 000200          3311#
MINHI1= 000000          4216#
MINLO1= 057467          4215#
MMVEC  = 000250          2155#   4768*   4769*
MODP15  034462          5290    5369    7872#
MPERHD  031366          4747    7187#
MSP    = 000100          3310#
MULHDL  023614          6166    6218#
MULSEL  012122          4553#   6733
MULTI1  023702          6085    6117    6170    6179    6231#
MXWRDC  035236          5737    5816    5905    8005    8024#
M.CADD= 017760          3382#
M.CDIF= 017760          3381#
M.DPID  055236         11165#
M.DPIM  055176         11136#
M.DP01  055206         11139#  11148
M.DP02  055226         11143   11147#
M.DP40  055274         11176#  11197
M.DP41  055330         11181   11188#
M.DP42  055340         11187   11192#
M.DP44  055372         11199   11204#
M.DP50  055404         11174   11208#
M.DRV  = 000007          3378#
M.HEAD= 007000          3385#
M.ID   = 000003          3380#
M.PAR  = 100000          3379#
M.SECT= 000760          3384#
M.SER  = 077770          3383#
NCYL1   005512          3715#
NCYL2   005514          3716#
NED    = 010000          3259#   6924    9467    9477    10047   10125   10468   10838
NEDHDL  030320          6651    6842    6924#   7450    7545
NEM    = 004000          3258#   9462   10062   10228   10468
NEWDRV  017272          5412#   5422    6246    9202    9716
NEWMSG  007434          4291#   7072
NEWON   005530          3722#   6628*   6652*   6697    6699    6713    6873*   6876    6936*   7451*   7455    7546*   7550
NEWPAS  017260          4692    4778    5409#   6283
NGOFST  005550          3729#   5592*   5624*   5653    5657
NMTSTS= 000006          3790#   5534    5684    5723    5800    5888    5989    6072
NOCHK  = 000400          3445#  10702   10764   10805   10832   10850   10871   10894   10903   10937
NODRLK  012010          4540#   7579*
NODRTS  007564          4309#   4896
NODSC  = 004000          3450#   9678   10290
NOFALS  011170          4458#   4895    5648    5655    8843
NONEXD  011703          4527#   7555
NORTRY  003142          3666#   6011*   7373*   9739
NOTALD  010440          4392#   5162    5167
NOTDFT  016554          5246    5263#
NOTLOK  012000          4538#   7580
NOTMSG  065636          4758   12603#
NOTRDY  011551          4510#   7563
NOTYPE  003146          3671#   4756    4759*
```

```
NTREDY  007506          4300#  7478
NULINP  015530          5063   5078#
NXDRIV  007472          4298#  7462
NXF   = 000004          3270#
OCTBIN  054456          5108   5282   5362   5402   7087   7923  10968#
OFFSET= 000115          3198#  5597   5606  10747
OFINUS  005516          3717#
OFSHED  011177          4463#  5579
OFST  = 000004          3290#
ONLINE  005526          3721#  6627*  6699   6712   6713*
OPCOMP  003122          3650#
OPCONT  003131          3657#
OPI   = 020000          3282#  5691   8817   9579  10068  10234  10470
OPIERR= 000010          4231#  9581   9734
OPNLST  016444          5149   5239#  5244   5251
OR    = 000200          3254#
OREQSW= 000020          3676#
OVLODR  007131          4255#
O.WAIT  003054          3594#  9968   9972*  10082* 10086  10102* 10168* 10181* 10196* 10740* 10800* 10862* 10879*
PACK  = 000103          3193#  7491   7585
PARMDE  010321          4376#  5133
PARM0   002630          3483#  3638   7372   7644   8962   8975   9691   9760   9811   9845
PARM1   002714          3515#  7378*  7381*  9435   9795   9829
PART  = 000001          1990#  2003   2009   2030   2039   2203   2207   2222   3693   3736   3771   3782   3797
                        3799   3803   3804   3806   3812   3814   3818   3819   3821   3830   3834   3842   3844
                        3848   3855   3857   3861   3862   3864   4243   4461   4462   4695   4753   4783   5169
                        5499   5502   6285   7285   7304   7525   8104   8151   8464   8532   8543   9089  12601
                       12650  12655
PART1   007071          4248#
PART2   007075          4249#  4755   6335
PAT   = 000020          3308#
PATRN   005536          3725#  5734*  5811*  5900*  5902*  5924   8266   8337   8444   8968
PAT00   006036          3869#  8268   8283   8339   8354
PAT01   006076          3891#
PAT02   006136          3913#
PAT03   006176          3935#
PAT04   006236          3957#
PAT05   006276          3979#
PAT06   006336          4000#
PAT07   006376          4022#
PAT08   006436          4043#
PAT09   006476          4065#
PAT10   006536          4087#
PAT11   006576          4109#
PAT12   006636          4131#
PAT13   006676          4153#  6006   6126
PAT14   006736          4175#  8195
PAT15   006776          4195#  6003   7850   7928*  7934*
PBLKT   003112          3638#  9965  10052  10078  10246  10271  10689*
PBSVAL= 001000          3446#  10627
PCA   = 004000          3315#
PCD   = 010000          3316#
PCLKF   003132          3658#
PCVEC   003164          3685#
PFIFTN  010616          4413#  5209
PGE   = 002000          3257#  9525  10468
```

```
PIP    = 020000          3300#
PIRQ   = 177772          2056#
PIRQVE= 000240           2150#
PKRB     003160          3683#
PKS      003154          3681#
PKSB     003156          3682#
PLOFST   005546          3728#  5591*  5626*  5646   5650
PMA      005604          3740#  5752*  5753*  5831*  5832*  5920*  5921*  7238   7240   8285   8356   8436   8438
                         8556   8598   8665   8667   8942*  8943*  8944*  8945   8946   8966*  8967*  8970*  8971*
PRDFLT   005706          3809#  5202   5253   5256   5295   5336   5432*  5444*  6432*  6443*  6660*  6671*  6850*
                         6861*
PREPAR   031606          5754   5833   5922   7236#  8951   9009
PREPKB   030536          5036   5183   5195   5220   5537   5726   5803   5891   5992   6075   6379   6393   6415
                         6466   6494   6553   6596   6624   6648   6802   6820   6888   6905   6989#  7622   8509
PRMBUF   013256          4676#  7817*  7818
PRMINP   007311          4274#  4806
PRMLIM   005734          3825#  5429*  5433*  5441*  5445*  6430*  6433*  6441*  6444*  6657*  6661*  6668*  6672*
                         6847*  6851*  6858*  6862*  7981   7983   7991   7994   7996   7999
PRMLST   005660          3794#  5201   5254   5294   5335   7819   7823   7986*  8002*  8003*
PRMNEM   006010          3852#  5196   5275   5331   7817   7821   7877   7974
PRMPHO   005256          3702#  9146*  12336
PRMPLO   005254          3701#  9147*  12336
PRMPSP   013277          4680#  5316
PROMPT   013274          4679#  4919   5083   5134   5267   7900
PRVCMD   005224          3697#  4772   6352   9135   9402
PR0    = 000000          2073#  4771   5412   6343   7647   9761   9833
PR1    = 000040          2074#
PR2    = 000100          2075#
PR3    = 000140          2076#
PR4    = 000200          2077#  4764   6339
PR5    = 000240          2078#  3570
PR6    = 000300          2079#
PR7    = 000340          2080#  4699   4748   4769   6286   6342   7151   7193   11592  11593
PS     = 177776          2053#  2054   4699*  4771*  5412*  6286*  6343*  7647*  9954   9960*  9974*  10683  10684*
                         10943*
PSTART   013336          2206   4696#
PSW    = 177776          2054#  9761*  9833*
PT       005700          3803#  5204   5900   6004   7880
PWRMSG   057100          11595  11599#
PWRVEC= 000024           2145#  4714*  4715*  6301*  6302*  11583*  11591*  11592*
P.ASOF= 000032           3467#  9389   10296*  10297  10330*  10561*  10749*  10750
P.A00  = 000040          3470#  7471   7481   7558   7575   9392   9801   10287*  10288  10304*  10305  10618*  10826*
P.A01  = 000044          3472#  6749   6772   7619   9394   10597*
P.A10  = 000050          3474#  9396   10604*
P.A11  = 000054          3476#  8054   9398   10611*
P.BAHI= 000007           3429#  5748*  5827*  5916*  7262*  8948*  9378
P.BALO= 000010           3430#  5745*  5824*  5913*  6009*  6021*  6126*  7261*  7502*  8091*  8528*  8785*  8797*
                         8945*  9381   10783  10880
P.BAR  = 000024          3464#  9388   10327*  10557*
P.B00  = 000042          3471#  9393   10282*  10283  10619*  10827*
P.B01  = 000046          3473#  9395   10598*
P.B10    000052          3475#  6190   9397   9823   10605*
P.B11  = 000056          3477#  9399   10203*  10612*
P.CMND= 000001           3423#  5460*  5594*  5597*  5599*  5604*  5606*  5675*  5760*  5764*  5840*  5844*  5935*
                         5939*  6012*  6018*  6023*  6100*  6128*  6163*  6171*  6175*  6226*  6500*  6505*  6510*
                         6552*  6563*  6570*  6575*  6580*  6683*  6737*  6747*  6751*  6768*  6770*  6774*  6783*
```

```
                    6872*  6886*  6896*  6901*  6908*  7375*  7438*  7449*  7491*  7493*  7495*  7544*  7570*
                    7583*  7585*  7596*  7615*  7617*  7629*  7645*  8050*  8088*  8178*  8492*  8531*  8536*
                    8776*  9369   9704*  9751*  9796*  9832*  9835*  10098  10715  10719  10731  10741  10747
                   10753  10756  10762  10787  10794  10813  10860  10877  10899  10908
P.CS1 = 000016       3461#  6221*  7396   7439*  9382   9500   9518   9864   9897   10119* 10171* 10209* 10210*
                   10212* 10213  10322* 10324* 10551* 10554* 10705  10759* 10760* 10762* 10766* 10767  10791*
                   10794* 10796* 10797* 10807* 10808  10816* 10817* 10819* 10820  10822* 10823  10828  10867*
                   10868* 10870* 10873* 10874  10890* 10891* 10893* 10896  10915* 10916  10925* 10926
P.CS1H= 000007        3428#  6155*  6157*  6209*  6211*  7379*  7380*  7381*  7504*  7505*  8084*  8775*  9843*
                    9844*  10209  10450  10759  10796  10816  10867  10881  10890
P.CS2 = 000020        3462#  5688   6924   9383   9457   9462   9467   9469   9475   9477   9525   9575   9659
                   10125  10325* 10555* 10713* 10738  10803* 10804  10815  10834  10838  10864* 10865  10889
P.CYLN= 000002        3424#  5543*  5553*  5556*  5557   5562*  5571   5573*  5581*  5637   5665   5667*  5673*
                    5994*  6103*  6120*  6151*  6165*  6555*  6890*  7498*  7500*  7597*  7599*  7602   7605
                    8089*  8245   8697*  8702*  8777*  8879   8933*  9371   9831*  10207  10743  10785  10887
P.DCYL= 000030        3466#  8883   8929   9384   9632   9854   9886   10329* 10563*
P.DRVN= 000000        3422#  6110*  6150*  6164*  6188*  6632*  6682*  6746*  6764*  6778*  6843*  7377*  7425*
                    7543*  10095  10272  10448  10687
P.DS  = 000036        3469#  9390   9510   9514   9565   10332* 10559* 10840
P.DTS = 000026        3465#  8885   8887   8930   8931   9385   9633   9635   9822   9857   9858   9887   9893
                   10328* 10558*
P.EPAT= 000062        3479#  9656   10121* 10707
P.EPOS= 000060        3478#  9655   10122*
P.ER  = 000034        3468#  5691   8817   9391   9456   10331* 10560*
P.OFST= 000006        3427#  5596*  5607*  10749
P.PRST= 000014        3432#  6001*  6127*  9483   9496   9668   9673   9678   9683   9798   9967*  10080* 10114*
                   10142* 10149  10153* 10250* 10273  10277* 10290* 10308* 10318* 10344  10348* 10358* 10370*
                   10475* 10627* 10702* 10739* 10763* 10764  10789  10795* 10801  10805  10832  10842* 10850
                   10871  10894  10903  10937
P.SECT= 000004        3425#  5544*  5566   5568*  5574*  5582*  5642   5661   5663*  5668*  5996*  6121*  8087*
                    8172   8183*  8222   8698*  8782*  8791*  8792   8798*  8808*  8809   8881   8935*  9374
                    9830*  10208  10744  10786  10888
P.TRCK= 000005        3426#  5545*  5559   5563*  5564   5569*  5580*  5631   5670*  5671   5995*  6526*  6792*
                    7501*  8090*  8173*  8175   8180*  8181   8236   8699*  8778*  8880   8934*  9372
P.WC  = 000012        3431#  5740*  5744*  5819*  5823*  5908*  5912*  5997*  6000*  6122*  6125*  7503*  8092*
                    8529*  8753   8784*  8938*  8964   9001   9376   10784
P.WCR = 000022        3463#  9386   10326* 10556*
QNEWSW= 000000        3182
Q.INIT= 040000        3453#
RCDASW= 000010        3675#
RCLREQ= 004000        4239#  9503   9508   9553   9558   9563   9568   9702
RDALHD= 000164        3211#  10098  10877
RDCHR = 104406        6363   7712   11894#
RDCHRS  034002        4863   4921   5014   5058   5085   5135   5241   5269   5318   5390   7073   7706#  7884
                    7902
RDDATA= 000121        3200#  5764   5844   5939   6023   7495   8088   8776
RDGATE= 100000        3319#
RDHDO   046514        9590   9615   9820#
RDHEAD= 000125        3202#  5460   9835   10212  10893
RDSTAT= 000141        3210#  6175   6747   6770   7449   7544   7617   8050   9796   10813
RDY   = 000200        3225#  6218   9698   10770  10823
RECAL = 000113        3197#  5675   6500   6505   6510   6563   6570   6575   6580   6783   6896   6901   6908
                    7493   7583   8492   9704   10756
RECODE  005474        3708#  6016   6169   7443*  7509   7642*  8399   8622   8994   8997   9075*  9083*  9171
                    9181*  9432*  9433   9460*  9465*  9486   9488   9503*  9508*  9548*  9553*  9558*  9563*
                    9568*  9573*  9581*  9585   9591   9606*  9610   9616   9631*  9638   9641*  9645*  9648*
```

```
                    9661* 9662* 9671* 9676* 9681* 9686* 9694  9700* 9702  9708  9710* 9734  9736*
                    9741  9748* 9768* 9800*
REDBSF  041066      5479  8773#
REFNEM  040716      6013  6024  8724#
REISSU  003137      3663# 9327*
RELEAS= 000140      3209# 10860
REPSUP  043676      6029  6186  6200  6222  7146  7190  7441  8336  8602  8794  9366# 9438
RESREG= 104410      7057  7138  7263  7317  8015  8073  8160  8184  8201  8248  8317  8398  8459
                    8538  8578  8621  8689  8764  8869  8894  8952  8980  9076  9126  9158  9298
                    9417  9763  9812  9846  9905  11240 11292 11896#
RESVEC= 000010      2140#
RETANL  046316      9436  9705  9767#
RETNML  046334      5695  6939  8821  9437  9770#
RKASOF= 000016      3178# 9789  10108 10178 10258 10513 10561 10750* 10775
RKBA  = 000004      3173# 9788  10509 10557 10783*
RKBADR  007337      4278# 4810
RKBAS   003036      3568# 4791* 4795* 4809  4812* 7371  9593  9618  9697  9750  9781  9961  10030
                    10694
RKCS1 = 000000      3171# 6734* 7467* 7565* 9594* 9598  9619* 9623  9783  10053 10105 10163 10213*
                    10222 10404* 10405 10414* 10453* 10454 10478* 10506 10767* 10769 10808* 10820* 10822
                    10852* 10874* 10896* 10905* 10915 10925 10940*
RKCS2 = 000010      3175# 7427* 7436  9784  10031 10191 10449* 10507 10555 10738* 10804* 10815* 10865*
                    10889* 10914*
RKDA  = 000006      3174# 9786  10208* 10510 10558 10744* 10786* 10888*
RKDB  = 000024      3181# 5471  5472  9595  9596  9597  9620  9621  9622  10187 10188 10189
RKDC  = 000020      3179# 9785
RKDCYL= 000020      3180# 10207* 10514 10563 10743* 10785* 10887*
RKDS  = 000012      3176# 9698  9790  10056 10511 10559
RKECPS= 000030      3186# 10122 10518
RKECPT= 000032      3188# 10121 10519
RKER  = 000014      3177# 7445  9791  10065 10231 10512 10560
RKMR1 = 000026      3182# 10199* 10278* 10515 10592* 10599* 10606* 10613* 10714*
RKMR2 = 000034      3183# 10516 10826
RKMR3 = 000036      3184# 10517 10827
RKPAT = 000032      3187#
RKPOS = 000030      3185#
RKPRI   003042      3570# 4767  4800* 4819  4835* 9960  10684
RKPRTY  007405      4286# 4824
RKVADR  007362      4282# 4815
RKVEC   003040      3569# 4765  4796* 4797* 4814  4817* 6340
RKWC  = 000002      3172# 9787  10508 10556 10784*
RLS   = 000010      3249# 10864
RNDADR  035634      6134  6554  6889  8141#
RNDLNG= 016514      4226# 6551  6836
RNDSHT= 002734      4225# 6832
RUNTST  017164      5169  5387#
RWBUF   065636      3738  5584  5729  5806  5894  6021  6022* 6027  6031  7203  7502  7512  8091
                    8096  8156  8158  8270  8341  8522  8528  8703  8966  12601#
RWNRDY  012610      4612# 7621
R.ABNL  052514      9973  10083 10158 10175 10353 10373 10633# 10843
R.CONT  052540      10035 10073 10137 10146 10217 10239 10262 10268 10301 10341 10365 10410 10462
                    10485 10623 10641# 10857 10911 10920 10933
R.NORM  052526      10111 10204 10313 10637# 10780 10853 10941
SAVPAR  003176      3691# 7250* 8289  8360  8561  8607
SAVPRS  005552      3730# 8963  8974
SAVREG= 104407      7018  7124  7236  7296  7299  7971  8048  8155  8171  8193  8216  8264  8331
```

		8393	8520	8549	8553	8591	8595	8616	8751	8831	8878	8927	8961	8973
		9065	9109	9134	9193	9311	9367	9429	9780	9821	9885	11225	11266	11895#
SAVWRD	003200	3692#	6184	6224*	6225*									
SAVXDP	032002	5731	5808	5896	7296#									
SCLR =	000040	3252#	10914											
SCNDRV	032404	4982	6677	6867	7423#									
SCOPER=	104411	5767	5942	11897#										
SCOPE1	057122	11610#	11897											
SCRACH	005534	3724#	4885*	4886*	4887	6629*	6632	6744	6746	6753*	6832*	6836*	6887	7433*
		7434*	7747*	7749*	7750	7760*	7761*	7762	8335*	8401	8414*	8417	8599*	8630
		8643*	8646											
SECNL1	010427	4390#	5157											
SECNL2	010457	4395#	5161											
SEEK =	000117	3199#	5594	6100	6128	6552	6886	7596	10731	10741				
SELDRV=	000101	3192#	6683	6872	7438	10870								
SELECT	005524	3720#	5813*	5834	5856*	6626*	6741	6754*						
SELP15	010645	4417#	7896											
SERNM	011110	4449#	8053	8095										
SETPRM	016730	5152	5315#	5321	5326	5370								
SETTUP	033640	5532	5721	5757	5798	5886	5932	5987	6070	7640#				
SETUPM	040612	5736	5814	5904	8697#									
SIZMEM	030570	4783	7018#											
SKI =	000002	3268#	9550											
SOFTBS	011136	4453#	8867											
SOR7	012535	4603#	6378											
SPACE1	013253	4674#	5082	5266	5645	5652	7791	7899						
SPACE2	013252	4673#	5641	8848	8854	9240	9244	9247	9250	9270	9275	9290		
SPACE3	013251	4672#	5636											
SPACE4	013250	4671#												
SPACE6	013246	4670#	7798											
SPAR =	020000	3232#	9500	10466										
SPDLSS=	000020	3292#	9514											
SRHTBS	042242	9026#	9067											
SRTSPL=	000111	3196#	6751	7629	10753									
SR0 =	177572	2159#	7137*	7147	7341*	7343*	7345*	7347*	8290*	8305*	8361*	8457*	8562*	8577*
		8608*	8688*	11765	11780*	11787*								
SR1 =	177574	2160#	7148											
SR2 =	177576	2161#	7149											
SR3 =	172516	2162#	5774*	5862*	5950*	7136*	7152	11777*						
ST	005704	3805#	5477											
STACK =	001100	2044#	4706	4859	5413	6293	7640*	7641	8491	11594				
STALL	035552	8112#	9110											
STALLS	005502	3711#	4777*	4860*	5131*	5477*	6351*	8121						
STKLMT=	177774	2055#												
STPASS	017246	5261	5407#											
STRCMD	042570	6152	8727	9114	9134#									
SUBCLR=	000177	3213#	5604	6226	6737	6774	7375	7570	7645	9751	9832	10908		
SUBMOD	025264	6463#	6485	6512	6566	6582								
SUBVER	007050	4245#	4754	6334										
SUMHI1	003174	3690#	8008*											
SUMLO1	003172	3689#	8007*	8009										
SVAL =	100000	3302#												
SVPRMS	042020	5758	5837	5933	8961#	9006								
SWR	001140	2274#	4704	4726*	4728	4734*	4741*	6291	6313*	6315	6321*	6328*	6336*	6337*
		7104	8114	8415	8442	8532	8644	8671	9203	9230	9258	11500	11507	11519
		11522	11612	11635	11653	11660								

```
SWREG   000176          2200#   4734    6321    6336    7104    7106    7109*
SWRMSG  007425          4289#   7107
SW0   = 000001          2108#
SW00  = 000001          2098#   2108
SW01  = 000002          2097#   2107
SW02  = 000004          2096#   2106
SW03  = 000010          2095#   2105
SW04  = 000020          2094#   2104
SW05  = 000040          2093#   2103
SW06  = 000100          2092#   2102
SW07  = 000200          2091#   2101
SW08  = 000400          2090#   2100
SW09  = 001000          2089#   2099
SW1   = 000002          2107#
SW10  = 002000          2088#
SW11  = 004000          2087#
SW12  = 010000          2086#
SW13  = 020000          2085#   9203
SW14  = 040000          2084#
SW15  = 100000          2083#
SW2   = 000004          2106#
SW3   = 000010          2105#
SW4   = 000020          2104#
SW5   = 000040          2103#
SW6   = 000100          2102#
SW7   = 000200          2101#
SW8   = 000400          2100#
SW9   = 001000          2099#
S.ACLO= 000100          3337#
S.BRHM= 000100          3352#
S.BRKE= 040000          3374#
S.CART= 000400          3354#
S.DCLO= 010000          3344#
S.DIB = 002000          3370#
S.DOOR= 000200          3353#
S.DRA = 000040          3323#
S.DROT= 020000          3345#
S.DRY = 000200          3325#   7471    7558
S.DSC = 040000          3332#   10140   10288   10305   10368
S.FLT = 000200          3338#   10283
S.FORM= 001000          3327#
S.FWD = 002000          3356#
S.HDFL= 000200          3367#
S.HDHM= 000040          3351#   6749    6772    7619
S.ICYL= 000040          3336#
S.ILF = 000400          3339#
S.LIMD= 020000          3373#
S.LOAD= 010000          3358#
S.MHD = 000400          3368#
S.NMOV= 010000          3372#
S.OFF = 002000          3328#
S.PAR = 001000          3340#   10143   10620
S.PIP = 020000          3331#   10151   10346
S.PLO = 004000          3371#
S.REV = 004000          3357#
S.RTZ = 020000          3359#
```

```
S.SECT= 000020           3364#
S.SKI = 002000           3341#
S.SPIN= 010000           3330#
S.SPLS= 010000           3343#
S.SPOK= 001000           3355#
S.TYPE= 000400           3326#
S.UNLD= 000000           3360#
S.UNS = 040000           3346#
S.VV  = 000100           3324#
S.WCLK= 000040           3365#
S.WGAT= 000100           3366#
S.WLE = 004000           3342#
S.WRL = 004000           3329#  7481   7575
S.XDOK= 000020           3350#
S.XERR= 001000           3369#
S0      005670           3799#  5155   5475   5681
S1      005672           3800#  5155   5476   5682
S2      005674           3801#  5159   5469   5677
S3      005676           3802#  5159   5470   5678
TBITVE= 000014           2141#
TCONHI  003170           3688#
TCONLO  003166           3687#
TELDRV  014460           4868   4876#  4924   4935   4990
TKVEC = 000060           2148#  4763*  4764*  6338*  6339*
TLSTHD  010143           4355#  5034   5079
TOPROC  046344           9314   9485   9779#
TPVEC = 000064           2149#
TRACK   005520           3718#  6518*  6526   6529   6762*  6786   6789*  6791*  6792   6795   6799*  6800
TRAPVE= 000034           2147#  4712*  4713*  6299*  6300*  11593*
TRKCHK  035716           8171#
TRKNLW  010471           4397#  5166
TRNSFR  042132           5761   5765   5841   5845   5936   5940   8990#
TRTVEC= 000014           2142#
TSTDRN  011051           4442#  5482
TSTING  003120           3648#  4770*  5408*  6131*  6174*  6350*  8474
TSTLST  005630           3770#  3790   5050   5071   5093   5113*  5118   5122*  7675   7799
TSTMDS  007737           4329#  5012
TSTMSG  060334           9224   11899#
TSTTYP  023564           6111   6168   6189   6207#
TST1    020010           5491   5530#
TST2    021136           5534   5684   5719#
TST3    021446           5723   5796#
TST4    022006           5800   5884#
TST5    022356           5888   5985#
TST6    022662           5989   6068#
TWOTOS= 002000           4238#  9486   9800
TYPAUT  012660           4620#  6593
TYPBSF  041356           5489   8831#
TYPDS = 104405           5651   5658   6261   11891#
TYPE  = 104401           4753   4754   4755   4758   4762   4806   4810   4815   4824   4836   4838   4839   4862
                         4872   4873   4876   4884   4887   4895   4896   4905   4908   4919   4933   4934   5012
                         5013   5020   5021   5034   5038   5057   5066   5067   5079   5082   5083   5125   5126
                         5130   5133   5134   5141   5142   5157   5161   5162   5166   5167   5185   5209   5222
                         5240   5249   5250   5266   5267   5298   5299   5316   5324   5325   5389   5398   5399
                         54?2   5579   5636   5641   5645   5648   5652   5655   5659   6259   6262   6333   6334
                         63.    6358   6362   6368   6370   6371   6373   6375   6378   6392   6413   6414   6465
```

		6491	6493	6528	6532	6539	6550	6565	6584	6593	6595	6621	6623	6733
		6758	6794	6798	6821	6885	6917	6957	6963	6967	6971	6973	6974	7005
		7006	7051	7055	7056	7072	7082	7083	7107	7461	7462	7477	7478	7487
		7488	7511	7516	7517	7555	7563	7580	7621	7717	7723	7729	7735	7746
		7750	7754	7762	7767	7769	7770	7772	7791	7798	7818	7843	7849	7883
		7892	7893	7896	7899	7900	7938	7939	8006	8012	8052	8053	8072	8094
		8095	8099	8502	8503	8834	8835	8836	8837	8843	8844	8848	8854	8858
		8867	9201	9217	9218	9219	9223	9224	9229	9232	9233	9234	9235	9240
		9243	9244	9245	9246	9247	9250	9253	9254	9256	9270	9275	9279	9285
		9290	9291	9293	11086	11333	11398	11473	11502	11510	11595	11887#		
TYPED	017356	5425#												
TYPERR	042760	9193#	11509											
TYPFMT	003124	3652#	4986	5427*	5439*	6428*	6438	6655*	6665	6845*	6855	7380	7423*	7428
		7447*	7496	7505	7600									
TYPLST	016172	5146	5181#											
TYPMAN	011466	4500#	6413											
TYPMOD	011425	4494#	6392											
TYPOC =	104402	6377	8847	8853	8857	9239	9249	9252	9272	11888#				
TYPON =	104404	5644	11890#											
TYPOS =	104403	5633	5638	7069	7795	7800	7846	7851	9226	11889#				
TYPPAT	034430	5221	7842#	7898										
TYPPRM	034350	5184	5265	7817#										
TYPTST	034314	5037	5081	7791#										
T.ASOF	003014	3555#	6224	10108*	10109	10133	10178*	10179	10258*	10259	10265	10296	10330	10338
		10362	10513*	10775*	10776									
T.BA	003006	3552#	10327	10509*										
T.CS1	003000	3547#	6218	6221	10053*	10105*	10106	10119	10130*	10163*	10165	10171	10200*	10222*
		10224	10254	10279*	10293*	10323*	10324	10335*	10359*	10405*	10406	10446	10450*	10451*
		10453	10457	10466	10506*	10553*	10554	10594*	10601*	10608*	10615*	10769*	10770	10772
T.CS2	003002	3549#	10031*	10032	10047	10050	10062	10091	10191*	10192	10228	10242	10325	10447*
		10448*	10449	10459	10468	10507*								
T.DA	003010	3553#	10328	10510*										
T.DB	003034	3564#												
T.DC	003012	3554#	10329	10514*										
T.DS	003020	3558#	10056*	10057	10332	10511*								
T.ER	003016	3557#	10065*	10068	10231*	10234	10331	10470	10512*					
T.MR1	003022	3559#	10515*											
T.MR2	003024	3560#	6625*	10140	10151	10287	10304	10346	10368	10516*	10597	10604	10611	10618
T.MR3	003026	3561#	10143	10203	10282	10517*	10598	10605	10612	10619	10620			
T.PAT	003032	3563#	10519*											
T.POS	003030	3562#	10518*											
T.WCR	003004	3551#	10326	10508*										
UBMPRS	003143	3667#	5772	5860	5948	7049*	7134	7251	8403	8632	9144			
UEXATT=	000010	3439#	9683	10114										
UFE =	000400	3255#	9469	9475	10047	10125	10468	10838						
UNLOAD=	000107	3195#	6768	7615	10719									
UNLOD	011322	4481#	6761*	6779*										
UNS =	040000	3283#	9545											
UPE =	020000	3260#	9457	10062	10228	10468								
VERIFY	011323	4482#	6464*	6523	6538*	6594*	6610*	6619	6766	6781				
VFHDSW=	000002	3673#												
VV =	000100	3295#												
WAIT4R	033534	6525	6546	7615#										
WCDASW=	000004	3674#												
WCE =	040000	3261#	5688	9659	10062	10228	10468							
WCEFLG	003140	3664#	5601*	5609	5694*	8787*	8789	8804*	8806	8820*	8992*	8996*		

```
WCERR = 000100          4234#  8994   9662   9734
WC2BIG  010503          4399#
WDSXFR  005564          3731#  8265   8334   8390   8555   8597   8613   8893*  8936   8964*  8965*
WHDSW = 000001          3672#
WLE   = 004000          3280#  9541  10068  10234
WORDSP  013262          4677#  7843
WRCKHD  021102          5600   5688#
WRDATA= 000123          3201#  5760   5840   5935   6012   6163   8531  10791
WRHEAD= 000127          3203#
WRL   = 004000          3299#
WRTCHK= 000131          3204#  5599   6018   6171   8178   8536  10787
WRTGAT= 040000          3318#
WRTLOK  007520          4302#  7488
WRTSEC  037562          5551   5555   5561   8520#
W.DRV   003114          3644#  9962*  10077* 10103* 10155* 10170* 10183* 10197* 10245* 10312* 10321* 10350* 10372*
                       10643*  10692* 10755* 10758* 10779* 10831* 10849*
W.MILI  003060          3597#  9957   10923
W.MIN   003066          3613#  10755
W.MTIM  003056          3596#  9955*  9957*
W.SEC   003062          3610#  10692
W.TIME  003100          3619#  9958   9964*  10076* 10104* 10154* 10169* 10182* 10198* 10244* 10311* 10320* 10349*
                       10371*  10642* 10691* 10778* 10830* 10848*
W.WCK = 000200          3444#  10702  10763  10789  10795
W.WTCH  047132          8732   9123   9950#  10768  10774  10821
W.8SEC  003064          3612#  10155  10350  10758
XDPSAV  005542          3727#  5729*  5730*  5806*  5807*  5894*  5895*  7308   7309   7335
XDPSVD  003134          3660#  4784   4787*  5733*  5810*  5898*  7301   8489*
XDP1    032030          7298   7304#
XDP2    032076          7302   7305   7317#  7324   7355
XDP3    032122          7311   7326#
XDP4    032102          7313   7319#
XOVLAD  003133          3659#  5732*  5809*  5897*  7300*  8483
XXDPAD  005540          3726#  7274*  7278*  7279*  7280*  7307
XXDPCH  003117          3647#
XXDPME  007166          4260#  4908
$APTHD  001000          2231   2237#
$ASTAT= ****** U        11711  11726
$ATYC   057430          11682  11684#
$ATY1   057404          11680#
$ATY3   057412          11071  11681#
$ATY4   057422          11515  11683#
$AUTOB  001134          2271#
$BASE   001374          2378#  4791
$BDADR  001122          2266#
$BDDAT  001126          2268#
$BELL   001310          2329#  6565   6917   11502  11534
$CHARC  055172          11088* 11098* 11105  11114* 11119#
$CKSWR= ****** U        11894
$CMTAG  001100          2254#  4701   4702   4710   4716   4717   4718   6288   6289   6297   6303   6304   6305
$CM1  = 000040          2286#  2287#  2288#  2289#  2290#  2291#  2292#  2293#  2294#  2295#  2296#  2297#  2298#
                        2299#  2300#  2301#  2302#  2303#  2304#  2305#  2306#  2307#  2308#  2309#  2310#  2311#
                        2312#  2313#  2314#  2315#  2316#  2317#  2318#
$CM2  = 000100          2286#  2287#  2288#  2289#  2290#  2291#  2292#  2293#  2294#  2295#  2296#  2297#  2298#
                        2299#  2300#  2301#  2302#  2303#  2304#  2305#  2306#  2307#  2308#  2309#  2310#  2311#
                        2312#  2313#  2314#  2315#  2316#  2317#  2318#
$CM3  = 000040          2284#  2286
```

```
$CM4   = 000011            2318#   2319#   2320#   2321#   2322#   2323#   2324#   2325#   2326#   2327#
$CNTLG  056763            11572#
$CNTLU  056756            11571#
$CORE   060060            11763   11790#
$CPUOP  001346             2352#
$CRLF   001315             2331#   4836    4905    5038    5185    5222    5659    6370    6373    6974    7055    7056    7767
                           7772    8012    8072    8099    8837    8844    8858    9223    9233    9235    9243    9246    9256
                           9279    9285    9293   11087   11122   11510   11534
$CROUT  060110            11790   11797#
$DBLK   056432            11439   11473   11481#
$DB2D   055534            11266#
$DB2O   055414             7053    7825    9010    8097   11225#
$DECVL  055714            11268   11314#
$DEVCT  001330             2343#   5410*   6245*
$DEVM   001376             2379#   4964
$DOAGN  024016             6255    6264    6270#
$DTBL   056422            11442   11477#
$ENDAD  024006             2217    6266#  11529
$ENDCT  023754             4716    6257#   6303
$ENDMG  024025             6259    6274#
$ENULL  024022             6262    6273#
$ENV    001340             2348#   6277   11066   11512   11689   11713
$ENVM   001341             2349#   4739    4792    4965    6326    7463   11068   11073   11691
$EOP    023702             6072    6243#
$EOPCT  023746             4716*   5260*   5404*   6254#   6258    6303*
$ERFLG  001103             2257#  11497*  11534   11610   11624   11649   11651   11657*  11677
$ERMAX  001115             2263#   4719*   4750*   6306*   7663*  11651   11672*  11677
$ERROR  056442             4710    6297   11496#
$ERRPC  001116             2264#  11504*  11505*  11506   11534   12321
$ERRTB  001400             2396#   9214
$ERTTL  001112             2261#  11503*  11534
$ESCAP  001306             2328#   4718*   6305*  11525   11527   11534   11671*
$ETABL  001340             2347#
$ETEND  001400             2243    2380#
$FATAL  001322             2340#  11717*
$FFLG   057650            11680*  11683*  11711   11720*  11728#
$FILLC  001156             2282#  11091   11122
$FILLS  001155             2281#  11122
$GDADR  001120             2265#
$GDDAT  001124             2267#
$GET42  023776             6263#
$GTSWR= ****** U          11893
$HD    = 000000            2018
$HIBTS  001000             2238#
$HINUM  054710             4781*   6347*   8198   11023   11031   11036*  11041#
$HIOCT  054610             5285    5365    7090    7926   10993*  11006#
$ICNT   001104             2258#  11662*  11663   11665*  11676
$INTAG  001135             2272#
$ITEMB  001114             2262#   9209   11506*  11514   11534
$KTNEX  060052            11764   11789#
$KTOUT  060042            11781   11786#
$KT11   057710             5749    5828    5917    7019*   7021    7310    8286    8357    8446    8455    8557    8567    8575
                           8603    8624    8677    8686    8949    9007   11762#  11766*  11789*
$LF     001316             2332#  11122   11534
$LFLG   057647            11721*  11727#
$LONUM  054712             4782*   6348*   8117    8142    8197   11022   11029   11035*  11042#
```

```
$LPADR  001106          2259#  4720*  6307*  11597  11655*  11669*  11674  11676
$LPERR  001110          2260#  4721*  5756*  5931*  6308*  11524  11614  11655  11670*  11676
$LSTAD  060154          7023  7036  11802*  11809#
$LSTBK  060156          7027  11803*  11810#
$MADR1  001352          2365#
$MADR2  001356          2369#
$MADR3  001362          2372#
$MADR4  001366          2375#
$MAIL   001320          2239  2243  2338#  4738  5531  5720  5797  5885  5986  6069  6325  11066  11512
                       11668
$MAMS1  001350          2359#
$MAMS2  001354          2367#
$MAMS3  001360          2370#
$MAMS4  001364          2373#
$MBADR  001002          2239#
$MFLG   057646          11681*  11687  11722*  11726#
$MNEW   057001          11575#
$MSGAD  001334          2345#  11697*  11700
$MSGLG  001336          2346#  11702*
$MSGTY  001320          2339#  11695  11703*  11715  11719*
$MSWR   056770          11573#
$MTYP1  001351          2360#
$MTYP2  001355          2368#
$MTYP3  001361          2371#
$MTYP4  001365          2374#
$MXCNT  057402          11666  11676#
$NULL   001154          2280#  11093  11122
$NWTST= 000001          5502#  5504  5699#  5701  5779#  5781  5867#  5869  5955#  5957  6041#  6043
$OCNT   056212          11370*  11399*  11412#
$OCTVL  055516          8055  11227  11252#
$OMODE  056214          11365*  11369*  11374  11377*  11388*  11414#
$OVER   057366          11636  11656  11664  11673#
$PASS   001326          2342#  4738*  5407*  5483  6251*  6252*  6260  6273  6279  6325*  7668
$PASTM  001006          2241#
$PWRCT  057076          11588*  11589*  11598#
$PWRDN  057012          4714  6301  11583#  11591
$PWRUP  057024          11583  11588#
$QUES   001314          2330#  4839  4873  4934  5021  5067  5126  5142  5250  5299  5325  5399  7006
                       7083  7770  7893  7939  11122  11534
$RAND   054612          8116  8141  8196  11018#
$RDCHR  056636          11550#  11894
$RDDEC= ****** U        11895
$RDLIN= ****** U        11895
$RDOCT= ****** U        11895
$RDSZ = 000000          11571#
$REGAD  001160          2284#
$REG0   001162          2286#  9370  12321
$REG1   001164          2287#  12321
$REG10  001202          2294#  7152*  7197*  8389*  8390*  8391  8429*  8432  8612*  8613*  8614  8658*  8661
                       9595*  9620*  9655*  12325  12333
$REG11  001204          2295#  8430*  8659*  9596*  9621*  9656*  12325  12333
$REG12  001206          2296#  8431*  8660*  9597*  9622*  9649*  9651*  12325  12333
$REG13  001210          2297#  8433*  8435*  8437*  8438*  8662*  8664*  8666*  8667*  12325  12333
$REG14  001212          2298#  6198*  8432*  8434*  8436*  8661*  8663*  8665*  12328  12333
$REG15  001214          2299#  6199*  8439*  8440*  8668*  8669*  12328  12333
$REG16  001216          2300#  8368*  8629*  12329  12333
```

```
$REG17  001220          2301#  12329
$REG2   001166          2288#  12321
$REG20  001222          2302#  12329
$REG21  001224          2303#  12329
$REG22  001226          2304#  12329
$REG23  001230          2305#  12329
$REG24  001232          2306#  12329
$REG25  001234          2307#  12329
$REG26  001236          2308#   9237   9401
$REG27  001240          2309#
$REG3   001170          2289#  12321
$REG30  001242          2310#
$REG31  001244          2311#
$REG32  001246          2312#
$REG33  001250          2313#
$REG34  001252          2314#
$REG35  001254          2315#
$REG36  001256          2316#   9380*  12324
$REG37  001260          2317#   9381*  12324
$REG4   001172          2290#
$REG5   001174          2291#   6030*  7147*  7191*  8405*  8411*  8419   8425*  8634*  8640*  8648   8654*  9177*
                        9178*   9377   9745*  9746*  9782   9854*  9874*  9886*  9900   9903   12325  12332
$REG6   001176          2292#   6031*  7148*  7195*  7203   7219*  8406*  8412*  8421   8426*  8635*  8641*  8650
                        8655*   9855*  9858*  9869   9871*  9873*  9893*  9894*  9895*  9896*  9899*  9901   9902
                       12325   12332
$REG7   001200          2293#   7149*  7196*  7201   7217*  8413*  8423   8427*  8642*  8652   8656*  9856*  9857*
                        9859   9862*  9866*  9868*  9900*  9902*  9904*  12325  12332
$RESRE  060216         11844#  11896
$RTNAD  024020          6272#
$R2A  = ****** U       11897
$SAVRE  060160         11828#  11895
$SCOPE  057146          4708   6295   11632#
$SETUP= 000037          4683#  4707   4708   4710   4712   4714   4716   4717   4718   4720   6249   6294   6295
                        6297   6299   6301   6303   6304   6305   6307   11497  11522  11529  11539  11577  11633
$SIZE   057652          7020   11750#
$SIZEX  060114         11788   11798#
$STUP = 177777          4683#
$SUPRS  055730          7054   7826   8011   8071   8098   11325#
$SVLAD  057332         11644   11667#
$SVPC = 000234          2215#  2220
$SWR  = 167000          1999#  2018   2022   2023   2024   2025   2026   2027   2028   2029   2327   2328   2329
                        4717   4718   4720   4721   5531   5720   5797   5885   5986   6069   6240   6250   6265
                        6271   6273   6304   6305   6307   6308   11488  11489  11490  11491  11492  11500  11507
                       11519  11522  11534  11625  11626  11627  11628  11629  11635  11647  11649  11650  11651
                       11658  11659  11660  11670  11673  11676
$SWREG  001342          2350#  4741   6328
$SWRMK= 000000         11629
$TESTN  001324          2341#  5531*  5720*  5797*  5885*  5986*  6069*  11668*
$TIMES  001304          2327#  4717*  5424*  6250*  6304*  7670*  7675*  8488*  11633  11658*  11663  11666*  11676
$TKB    001146          2277#  6947   6989*  11537  11554  11560
$TKS    001144          2276#  5053*  5206*  5226*  6248*  6562*  6916*  6958*  6991*  11537  11552  11558
$TMP0   001262          2318#
$TMP1   001264          2319#  9030*  9031*  9039
$TMP10  001302          2326#  9043*  9044*  9045
$TMP2   001266          2320#  9026*  9054
$TMP3   001270          2321#  9027*  9055
```

```
$TMP4   001272            2322#   9028*   9056
$TMP5   001274            2323#
$TMP6   001276            2324#
$TMP7   001300            2325#
$TN   = 000007            2008#   2018    5491    5502    5531#   5534    5683    5684    5699    5720#   5723    5779    5797#
                          5800    5867    5885#   5888    5955    5986#   5989    6041    6069#   6072
$TNPWR  055644           11273   11274   11294#
$TPB    001152            2279#  11*11*  11122
$TPFLG  001157            2283#  11060   11122
$TPS    001150            2278#  11109   11122
$TRAP   060254            4712    6299   11864#
$TRAP2  060276           11875#  11886
$TRP  = 000012           11879#  11888#  11889#  11890#  11891#  11892#  11894   11895#  11896#  11897#  11898#
$TRPAD  060310           11869   11886#
$TSTM   001004            2240#
$TSTNM  001102            2256#   5423*   6249*   6281*   7672    8494*   9225   11499   11534   11624   11667*  11668   11673
                         11677
$TYPBN= ****** U         11892
$TYPDS  056216           11427#  11891
$TYPE   054714           11060#  11708   11879   11887
$TYPEC  055126           11090   11097   11104   11109#  11110
$TYPEX  055174           11115   11117   11120#
$TYPOC  056014           11368#  11888
$TYPON  056030           11367   11370#  11890
$TYPOS  055770           11363#  11889
$UNIT   001332            2344#   5457*
$UNITM  001010            2242#
$USWR   001344            2351#
$VECT1  001370            2376#   4790
$VECT2  001372            2377#
$XTSTR  057164           11634   11638#
$$GET4= 000000            6265#
$.$.SM= ****** U          3596    9979   10214   10405   10416   10454   10768   10809   10821   10875   10897   10915   10942
                         10951
$OFILL  056213           11364*  11368*  11378   11413#
$40CAT= ****** U         11509   11635
.     = 070671            2194#   2198#   2203#   2205#   2207#   2215    2216#   2218#   2220#   2221#   2227    2228#   2230#
                          2232#   2253#   2333    3677#   3694#   3695#   3696#   3705#   3727#   3730#   3740#   4675#   4705
                          4720    4721    5756    5931    6273    6277    6292    6307    6308   11122   11194   11252#  11314#
                         11481#  11534   11537   11577   11585   11602#  11676   11677   11729#  12320#
.$ASTA= ****** U         11681   11684
.$X   = 001000            2227#   2232
..HIGH  000232            2210#
..LOW   000230            2209#
```

```
B4EOPS   6235#  6244
B4SCOP  11619#  11633
CKEXIT   2899#  8724   9111
COMMEN      1#  2151#
ENDCOM      1#  2151#
ERROR    2045#  6032   6187   6201   6223   7157   7200   7220   7442   7465   8407   8408   8410   8428   8441
         8636   8637   8639   8657   8670   8740   8795   9179   9317   9321   9328   9332   9336   9340   9344
         9348   9352   9356   9358   9459   9464   9471   9473   9479   9481   9490   9492   9494   9498   9502
         9507   9512   9516   9520   9523   9527   9531   9535   9539   9543   9547   9552   9557   9562   9567
         9572   9577   9587   9600   9602   9612   9625   9627   9640   9657   9666   9670   9675   9680   9685
         9711   9713   9747
ESCAPE      1#  2151#
GETPRI      1#  2151#  11758
GETSWR      1#  2151#
MSG      5502#  5504   5699#  5701   5779#  5781   5867#  5869   5955#  5957   6041#  6043
MULT        1#  2151#
NEWTST      1#  2151#  5502   5699   5779   5867   5955   6041
NEXTAD   2898#
NXTAD    2897#
NXTST    2895#  5534   5683   5723   5800   5888   5989   6072
PARMBK      1#  3480   3512
PARMDF      1#  3386
POP         1#  2151#  11037  11468  11723  11724  11849
PUSH        1#  2151#  11018  11427  11684  11686  11707  11829
P.DRVR      1#  3544
REGDEF      1#  3168
REPORT      1#  2151#
R.DRIV      1#  9908
R.QDRV      1#
SCOPE    2046#  5530   5719   5796   5884   5985   6068   6244
SETPRI      1#  2151#
SETTRA  11879#  11888  11889  11890  11891  11894  11895  11896  11897
SETUP       1#  2151#  4700   6287
SKIP        1#  2151#  5491
SLASH       1#  2151#
SPACE    2151#
STARS       1#  2000   2006   2151#  2213   2224   2226   2233   2249   2333   2336   4848   4856   4995   5009
         5502   5529   5699   5718   5779   5795   5867   5883   5955   5984   6041   6067   6237   6942   6944
         6979   6987   6996   7003   7011   7017   7062   7067   7097   7103   7114   7123   7143   7145   7162
         7164   7184   7186   7230   7235   7268   7271   7285   7295   7359   7367   7386   7391   7406   7421
         7525   7541   7593   7595   7611   7614   7635   7639   7652   7683   7704   7781   7789   7806   7815
         7831   7840   7858   7870   7955   7969   8020   8023   8043   8047   8078   8082   8104   8110   8134
         8140   8151   8154   8165   8170   8189   8192   8206   8215   8253   8262   8322   8330   8464   8473
         8514   8519   8543   8558   8583   8590   8694   8696   8714   8723   8748   8750   8769   8772   8825
         8830   8874   8877   8899   8903   8921   8926   8957   8960   8985   8989   9015   9025   9058   9064
         9091   9107   9131   9133   9163   9169   9183   9192   9300   9310   9361   9419   9427   9774   9778
         9817   9819   9851   9853   9879   9884   9917   9948   9982  10022  10385  10402  10420  10444  10491
        10504  10523  10549  10568  10590  10648  10675  10953  10966  11009  11045  11125  11135  11153  11164
        11216  11255  11317  11340  11417  11484  11536  11542  11606  11609  11621  11679  11736  11813  11858
SWRSU       1#  1999#  2151#  4722#  6309#
S.DRVE      1#  3320
TEST     2896#  5502   5699   5779   5867   5955   6041
TRMTRP  11879#
TYPBIN      1#  2151#
TYPDEC      1#  2151#  6260
TYPNAM      1#  2151#
```

```
TYPNUM      1#   2151#
TYPOCS      1#   2151#
TYPOCT      1#   2151#
TYPTXT      1#   2151#
$DECBN      1#
$OCTBN      1#  10951
$$CMRE   2247#   2286   2287   2288   2289   2290   2291   2292   2293   2294   2295   2296   2297   2298   2299
         2300    2301   2302   2303   2304   2305   2306   2307   2308   2309   2310   2311   2312   2313   2314
         2315    2316   2317
$$CMTM   2247#   2318   2319   2320   2321   2322   2323   2324   2325   2326
$$ESCA      1#   2151#
$$NEWT      1#   2151#  5502   5699   5779   5867   5955   6041
$$SET   11879#  11888  11889  11890  11891  11894  11895  11896  11897
$$SETM   4738#   6325#
$$SKIP      1#   2151#  5491
.EQUAT      1#   1999#  2041
.HEADE      1#   1999#  2009
.KT11       1#   1999#  2151
.SETUP      1#   1999#  4683
.SWRHI      1#   1999#  2018
.SWRLO   1999#   2030#  2031   2032
.$ACT1      1#   1999#  2211
.$APTB      1#   2334#
.$APTH      1#   1999#  2222
.$APTY      1#   1999#  11677
.$ASTA      1#
.$CATC      1#   1999#  2192
.$CMTA      1#   1999#  2247
.$DB2D      1#   1999#  11253
.$DB2O      1#   1999#  11214
.$DIV       1#
.$EOP       1#   1999#   6235
.$ERRO      1#   1999#  11482
.$ERRT      1#
.$MULT      1#
.$POWE      1#   1999#
.$RAND      1#   1999#  11007
.$RDDE      1#
.$RDOC      1#
.$READ      1#   1999#  11534
.$R2AZ      1#
.$SAVE      1#   1999#  11811
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#   1999#  11619
.$SIZE      1#   1999#  11734
.$SUPR      1#   1999#  11315
.$TRAP      1#   1999#  11856
.$TYPB      1#
.$TYPD      1#   1999#  11415
.$TYPE      1#   1999#  11043
.$TYPO      1#   1999#  11338
.$40CA      1#
.1170       1#
```