RK611/RK06

SUBSYSTEM VERIFY PART 1

MD-11-DZR6M-C

EP-DZR6M-C-DL-B

FICHE 1 OF 2

DEC 1976

digital

MADE IN USA

RK611/RK06

SUBSYSTEM VERIFY PART 1

MD-11-DZR6M-C

EP-DZR6M-C-DL-B

FICHE 2 OF 2

DEC 1976

digital

MADE IN USA

.REM @

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZR6M-C-D

PRODUCT NAME: RK611/RK06 SUBSYSTEM VERIFICATION: PART 1

DATE: DECEMBER 1976

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: DAVE HOFFMAN

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	HARDWARE REQUIREMENTS
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

TABLE OF CONTENTS (CONT'D)

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

9.0 DESCRIPTION OF TESTS
9.1 SEEK TESTS
9.1.1 TEST 1 - RECALIBRATE/SEEK TEST
9.1.2 TEST 2 - SEEK/SEEK TEST
9.1.3 TEST 3 - CYLINDER ADDRESSING TEST
9.1.4 TEST 4 - CYLINDER ADDRESS CROSSTALK TEST
9.1.5 TEST 5 - INCREMENT/DECREMENT SEEK TEST
9.1.6 TEST 6 - OSCILLATING SEEK TEST
9.1.7 TEST 7 - CONVERGING/DIVERGING SEEK TEST
9.1.8 TEST 10 - PSEUDO-RANDOM SEEK TEST
9.1.9 TEST 11 - MAXIMUM VELOCITY REVERSAL SEEK TEST
9.1.10 TEST 12 - MECHANICAL VIBRATION SEEK TEST
9.2 TIMING TESTS
9.2.1 TEST 13 - MAXIMUM ROTATIONAL LATENCY MEASUREMENT
9.2.2 TEST 14 - ONE CYLINDER SEEK TIME MEASUREMENT
9.2.3 TEST 15 - AVERAGE SEEK TIME MEASUREMENT
9.2.4 TEST 16 - MAXIMUM SEEK TIME MEASUREMENT
9.3 TEST 17 - SECTOR ADDRESSING TEST
9.4 TEST 20 - TRACK ADDRESSING TEST
9.5 TEST 21 - READ/WRITE DATA TEST
9.5.1 DATA PATTERNS
9.6 TEST 22 - DUAL-ACCESS DATA TEST

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

APPENDIX A SAMPLE ADDRESS 200 DEFAULT RUN

APPENDIX B SAMPLE ADDRESS 204 RUN

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 1 OF THE RK611/RK06 SUBSYSTEM VERIFICATION TESTS, WHICH PROVIDE A FUNCTIONAL SHAKEDOWN OF THE ENTIRE SUBSYSTEM, INCLUDING THE UNIBUS INTERFACE AND ACCESS TO MAIN MEMORY. THE TESTING IN PART 1 EMPLOYS WORST-CASE SITUATIONS INVOLVING MECHANICAL POSITIONING, DISK ADDRESSING, AND DATA TRANSFER. IN ADDITION, MEASUREMENTS ARE MADE PERTAINING TO DRIVE OPERATIONAL TIMING.

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

- RK611 REGISTER ADDRESS
- RK06 VECTOR ADDRESS
- RK06 PRIORITY LEVEL
- DRIVE(S) TO BE TESTED
- TEST(S) TO BE RUN
- NUMBER OF TEST ITERATIONS
- DISK ADDRESS LIMITS
- DISK ADDRESS INCREMENTS
- DATA PATTERNS USED
- PHYSICAL MEMORY ADDRESS ON DATA TRANSFERS
- WORD COUNT ON DATA TRANSFERS
- 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

THE FOLLOWING HARDWARE IS REQUIRED TO RUN PART 1 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
- KW11-L OR KW11-P CLOCK
- RK06 UNIBUS CONTROLLER (RK611)
- 1 TO 8 RK06 DRIVES
- 1 TO 8 RK06 DISK CARTRIDGES (FORMATTED IN 20 OR 22 SECTOR FORMAT)

NOTE

IF NEITHER KW11-L OR P CLOCK IS PROVIDED, THE TIMING TESTS IN SECTION

9.2 WILL BE BYPASSED, AND A MESSAGE WILL
INFORM THE OPERATOR.

3.0 PRELIMINARY PROGRAM REQUIREMENTS

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

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 SYSMAC

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

4.2 XXDP

THIS PROGRAM MAY BE LOADED UNDER XXDP. SUBSYSTEM TESTS 1-21 MAY BE RUN IN CHAIN OR DUMP MODE, BUT DUAL-ACCESS TEST 22 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. A MESSAGE WILL INFORM THE OPERATOR WHEN THIS IS NECESSARY.

4.3 ACT/APT

THE PROGRAM IS COMPATIBLE WITH ACT/APT CONVENTIONS. SUBSYSTEM TESTS 1-21 MAY BE RUN IN AUTOMATIC OR 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 1-21 OR DUAL-ACCESS DATA TEST 26 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 PROGRAMS :

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 BE 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.

12. REMAINING ETABLE ENTRIES ARE NOT USED.

4.4 DUAL-ACCESS

THIS PROGRAM IS DESIGNED TO UTILIZE DUAL-ACCESS IN TEST 22 ONLY (DUAL-ACCESS DATA TEST - SEE SECTION 9.6). FOR THE PURPOSES OF ALL OTHER TESTS (1-21). THE OPERATOR MUST GUARANTEE THAT THERE IS NO INTERFERENCE FROM THE UNUSED PORT. IF FAILURES ARE ENCOUNTERED IN TESTS 1-21 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 21 AND 22 ONLY. IN THESE TESTS, NPR TRANSFERS BETWEEN THE RK06 AND ANY SPECIFIED PHYSICAL MEMORY ABOVE THE PROGRAM, ARE PERFORMED, DURING READ/WRITE TESTING. (NOTE - TESTS 21 AND 22 ALSO SUPPORT 22-BIT ADDRESSING AND THE UNIBUS MAP, IF INSTALLED). IN ALL OTHER TESTS, 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 LESSER DEGREE, UPON THE PROCESSOR. HOWEVER, THE "AVERAGE" TIME REQUIRED TO RUN A QUICK-VERIFICATION (FIRST PASS) IS 14 MINUTES PER DRIVE. A SUBSEQUENT PASS WITH PARAMETERS DEFAULTED REQUIRES 21 MINUTES PER DRIVE.

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 1-21 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS,RK06 UNIBUS ADDR. AND INTERRUPT VECTOR FOR TESTS 1-21
220 DUAL-ACCESS DATA TEST 22 START ADDRESS

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
07	DO EXPLICIT SEEKS IN TESTS 1-12
06	REPORT ONE ERROR PER TRANSFER IN TESTS 17,21
05	INHIBIT WRITES IN TEST 21
04	INHIBIT WRITE CHECKS IN TEST 21
03	INHIBIT READS AND SOFTWARE COMPARES IN TEST 21
02	INHIBIT SOFTWARE COMPARES IN TEST 21
01	READ AFTER A WRITE CHECK ERROR IN TEST 21
00	REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 17,21

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: "DZR6M-C - RK611/RK06 SUBSYSTEM VERIFICATION: PART 1." FOLLOWED BY: "LAST PHYS MEM ADR=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 OPEPATING 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

IC — CYLINDER INCREMENT VALUE IN TESTS WHICH STEP THROUGH CYLINDER ADDRESSES, IC=1-632, DEFAULT=1

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

IT — TRACK INCREMENT VALUE IN TESTS WHICH ARE DONE OVER A RANGE OF TRACKS, IT=1-2, DEFAULT=1

SO — FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMATTED CARTRIDGE. SO=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 FORMATTED 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

IS — SECTOR INCREMENT VALUE IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, IS=1-25, DEFAULT=1

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=0-000070, DEFAULT=0

WC — WORD COUNT (NO. OF WORDS) USED IN DISK READ, WRITE, OR WRITE CHECK COMMANDS IN DATA TEST 20 , WC=0-177777.

DEFAULT=MAXIMUM BUFFER AVAILABLE, WC=0 IS INTERPRETED AS 65,536 WORDS

MA — PHYSICAL MEMORY ADDRESS (STARTING ADDRESS OF DATA BUFFER) USED ON ALL DATA TRANSFERS IN TESTS 21, 22. MA MAY BE GREATER THAN OR EQUAL TO ADDRESS OF RWBUF (NOT EXCEEDING AVAILABLE MEMORY ON THE SYSTEM). DEFAULT = ADDRESS OF RWBUF.

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

SM — MAXIMUM NUMBER OF UNIT STALL TIMES USED IN TEST 12 ONLY

8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)

8.2.1 DRIVE SELECTION

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

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

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

```
PARAMETER INPUT MODE

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

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

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

8.2.2.5 AND 9.2.3.6) THE PROGRAM FIRST TYPES:

```
     TO TEST AL 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. THE LIST OF DEFAULT ITERATION NUMBERS BEGINS AT ADDRESS "DFLTST", AND CAN BE ALTERED IN CORE.

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.

THE LIST OF DEFAULT PARAMETERS BEGINS AT ADDRESS "PRDFLT", AND CAN BE ALTERED IN CORE.

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 P), 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 IS, THE OPERATOR MAY TYPE (!) (EXCLAMATION POINT) AND THE PROGRAM WILL FILL UP THE REST OF THE PATTERN IS BUFFER WITH THE LAST WORD SPECIFIED. NEXT, THE PROGRAM WILL CONTINUE, IN THE PROPER PARAMETER INPUT MODE (O OR S).

8.2.4.2 CS- CONTROL SWITCH WORD

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

BIT	OPTION
05	DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
04	TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
03	INHIBIT TIMING REPORTS IN TESTS 13-16

NOTE

OTHER BITS UNUSED

9.0 DESCRIPTION OF TESTS

9.1 SEEK TESTS

THIS GROUP OF TESTS PERFORMS A VARIETY OF POSITIONING OPERATIONS. THROUGH THE EXECUTION OF READ HEADER COMMANDS, IMPLIED SEEKS ARE DONE TO SELECTED CYLINDERS, AND HEADER WORDS ARE READ AND CHECKED TO VERIFY THAT THE CORRECT CYLINDER WAS ADDRESSED. TESTING BEGINS WITH SIMPLE OPERATIONS WHICH VERIFY CYLINDER ADDRESSING CAPABILITY, AND PROCEEDS TO MORE INVOLVED SEEKING WHICH STRESSES THE SERVO MECHANISM. AT THE COMPLETION OF EACH SUBSYSTEM COMMAND, STATUS INDICATIONS AND ERROR BITS ARE CHECKED TO DETERMINE THE SUCCESS OF THE OPERATION. THROUGHOUT TESTING, SECTOR=FS, AND TRACK=FT.

9.1.1 TEST 1 - RECALIBRATE/SEEK TEST

THIS TEST WILL PERFORM A RECALIBRATE COMMAND (POSITION TO CYLINDER 0), FOLLOWED BY A SEEK TO CYLINDER LC. (AS IN ALL OF THE SEEK TESTS SEEKING IS IMPLIED, VIA THE READ HEADER COMMAND).

9.1.2 TEST 2 - SEEK/SEEK TEST

THIS TEST WILL PERFORM A SEEK TO CYLINDER FC, FOLLOWED BY A SEEK TO CYLINDER LC.

9.1.3 TEST 3 - CYLINDER ADDRESSING TEST

THIS TEST VERIFIES THE CYLINDER ADDRESS BITS, BY SEEKING TO CYLINDERS 0, 1, 2, 4, 10, 20, 40, 100, 200, AND 400 (OCTAL). THEN, THE SEEKS ARE DONE IN REVERSE BACK TO CYLINDER 0. IN BINARY, THE CYLINDER ADDRESSES SEQUENCE AS FOLLOWS:

```
000000000
000000001
000000010
    .
    .
    .
100000000
010000000
001000000
    .
    .
    .
000000001
000000000
```

9.1.4 TEST 4 - CYLINDER ADDRESS CROSSTALK TEST

THIS TEST PERFORMS SEEKS TO CYLINDERS WHOSE ADDRESSES ARE SUSCEPTIBLE TO BIT CROSSTALK WITHIN THE CYLINDER ADDRESSING LOGIC. FIRST A SEEK TO CYLINDER 0 IS DONE, FOLLOWED BY A SEEK TO 377 (OCTAL). THEN A SEEK TO 0 IS DONE AGAIN, AND A SEEK TO 376 , ETC. THE CYLINDER ADDRESSES SEQUENCE AS FOLLOWS:

```
000 000 000
011 111 111
000 000 000
011 111 110
000 000 000
011 111 101
000 000 000
011 111 011
    .
    .
    .
000 000 000
010 111 111
000 000 000
```

101 111 111

BY SEEKING TO 0 BETWEEN CROSSTALK SEEKS, THE CROSSTALK PATTERNS ALSO TEST THE CYLINDER DIFFERENCE LOGIC.

9.1.5 TEST 5 - INCREMENT/DECREMENT SEEK TEST

IN THIS TEST SEEKS ARE DONE IN INCREMENTS OF IC CYLINDERS STARTING AT CYL FC, AND ENDING AT OR BEYOND LC. THEN, THE SEEKS ARE DONE IN REVERSE, BACK TO FC. IF FC IS CHOSEN > LC, THE SEEKS WILL PROCEED IN THE OPPOSITE DIRECTION.

9.1.6 TEST 6 - OSCILLATING SEEK TEST

THIS TEST FIRST SEEKS TO CYLINDER FC, WHICH IS INITIALLY EQUAL TO NC. THEN NC IS INCREMENTED BY IC, AND A SEEK FROM FC TO NC IS MADE, FOLLOWED BY A SEEK BACK TO FC. NC IS INCREMENTED AGAIN, AND SEEKING FROM FC TO NC TO FC IS REPEATED UNTIL NC EXCEEDS LC. THEN, THE REVERSE IS DONE, DECREMENTING NC UNTIL NC EXCEEDS FC.

9.1.7 TEST 7 - CONVERGING/DIVERGING SEEK TEST

THIS TEST PERFORMS SEEKS WHICH EXERCISE CYLINDER DIFFERENCE VALUES. INITIALLY, NCYL1 = FC AND NCYL2 = LC. A SEEK IS DONE TO NCYL1 AND THEN TO NCYL2, THEN, NCYL1 IS INCREMENTED BY IC IN THE DIRECTION OF LC, AND NCYL2 IS INCREMENTED BY IC IN THE DIRECTION OF FC, AND SEEKS ARE DONE TO NCYL1 AND NCYL2 AGAIN. THIS SEEKING IS CONTINUED WITH INCREMENTED VALUES OF NCYL1 AND NCYL2, UNTIL NCYL1 EXCEEDS LC AND NCYL2 EXCEEDS FC. (NOTE: FC > LC IS PERMISSIBLE.) THIS TEST CAUSES SEEKING TO CONVERGING AND THEN DIVERGING CYLINDER VALUES.

9.1.8 TEST 10 - PSEUDO-RANDOM SEEK TEST

THIS TEST PERFORMS A SEEK TO A PSEUDO-RANDOMLY CHOSEN CYLINDER WHICH IS WITHIN THE RANGE (0-632).

9.1.9 TEST 11 - MAXIMUM VELOCITY REVERSAL SEEK TEST

THIS TEST PERFORMS A SEEK FROM CYLINDER 0 TO CYLINDER 201(OCT), AND BACK TO CYLINDER 0. THIS PARTICULAR SEEK CAUSES THE HEADS TO ACCELERATE TO MAXIMUM VELOCITY, AND THEN IMMEDIATELY DECELERATE, WITH NO APPRECIABLE PLATEAU OF CONSTANT VELOCITY. THIS OPERATION INDUCES HEATING IN THE SERVO MECHANISM.

9.1.10 TEST 12 - MECHANICAL VIBRATION SEEK TEST

THIS TEST PERFORMS GEOMETRICALLY INCREASING SEEKS WITH GEOMETRICALLY DECREASING STALL TIME BETWEEN OPERATIONS WITH INTENT TO INDUCE VARYING VIBRATIONAL MODES UPON THE DRIVE. THE TEST BEGINS WITH LC = 1, AND ST = SM, THEN, THE FOLLOWING SEQUENCE IS PERFORMED: SEEKS ARE DONE BETWEEN 0 AND LC, 10(DEC) TIMES. THEN, ST IS DIVIDED BY 2 AND LC IS DOUBLED, AND SEEKS ARE DONE BETWEEN 0 AND LC AGAIN, 10(DEC) TIMES. THIS PROCESS IS CONTINUED FOR NEW VALUES OF ST AND LC, UNTIL LC EXCEEDS CYL 400 (OCT). THEN, THE WHOLE PROCESS IS REVERSED, WITH ST BEING DOUBLED AND LC DIVIDED BY 2, UNTIL LC BECOMES < 1.

9.2 TIMING TESTS

THIS GROUP OF TESTS PERFORMS TIMING MEASUREMENTS OF BASIC DRIVE OPERATIONS - SPINDLE ROTATION AND SEEKING. FOR EACH TEST THE FUNCTION BEING MEASURED IS TIMED FOR A GIVEN NUMBER OF OCCURENCES, AND THE MINIMUM, MAXIMUM, AND AVERAGE VALUES ARE TYPED ALONG WITH THE NUMBER OF OPERATIONS WHICH EXCEEDED THE LIMITS SPECIFIED IN THE RK06 DISK DRIVE SPECIFICATION.

NOTE- FOR 50 HZ OPERATION, PATCH 1 INTO LOCATION 165 (LABELED HZ:).

9.2.1 TEST 13 - MAXIMUM ROTATIONAL LATENCY MEASUREMENT

THIS TEST MEASURES THE INTERVAL OF TIME BETWEEN 2 INDEX MARKS IN A SINGLE DISK ROTATION. THE SPECIFIED MAXIMUM ROTATIONAL LATENCY = 25 MILLI-SEC + OR - 2.5%. THIS MEASUREMENT IS MADE 128 TIMES.

SAMPLE PRINTOUT --

```
     ROTATIONAL TIMES :
     MIN = 24086 US  103 OF 128 BELOW SPEC'D MIN OF 24375 US
     MAX = 25065 US    0 OF 128 ABOVE SPEC'D MAX OF 25625 US
     AVG = 24285 US
```

9.2.2 TEST 14 - ONE CYLINDER SEEK TIME MEASUREMENT

THIS TEST MEASURES THE TIME REQUIRED TO SEEK BETWEEN 2 ADJACENT CYLINDERS, BOTH IN THE FORWARD AND REVERSE DIRECTIONS. SEEKS ARE DONE TO 0,1,2,...,631,632 AND THEN TO 631,630,...,2,1,0 AND THE RESULTS ARE TYPED FOR EACH DIRECTION. THE SPECIFIED ONE CYL SEEK TIME IS < 8 MILLI-SEC.

SAMPLE PRINTOUT --

```
     ONE CYL SEEK TIMES :
```

```
1039
1040
1041
1042                          **FORWARD DIRECTION**
1043                          MIN = 6332 US
1044                          MAX = 6749 US  0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1045                          AVG = 6536 US
1046                          **REVERSE DIRECTION**
1047                          MIN = 6314 US
1048                          MAX = 6749 US  0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1049                          AVG = 6532 US
1050
1051
1052
1053                    9.2.3  TEST 15 - AVERAGE SEEK TIME MEASUREMENT
1054
1055                    THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD  AND
1056                    REVERSE DIRECTIONS.  THE AVERAGE TIME IS CALCULATED FROM THE FOLLOWING
1057                    FORMULA :
1058                      T AVG = [T1(410)(2)+T2(409)(2)+...+T410(1)(2)]/(410)(410)
1059                    WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS.  FORWARD AND REVERSE
1060                    TIMES  ARE  MEASURED  AND TYPED, SEPARATELY.  THE AVERAGE SEEK TIME IS
1061                    SPECIFIED TO BE < 38 MILLI-SEC.
1062
1063                    SAMPLE PRINTOUT --
1064
1065                         AVERAGE SEEK TIMES :
1066                         **FORWARD DIRECTION**
1067                         AVG = 35673 US   SPEC'D MAX IS 38000 US
1068                         **REVERSE DIRECTION**
1069                         AVG = 35823 US   SPEC'D MAX IS 38000 US
1070
1071
1072
1073                    9.2.4  TEST 16 - MAXIMUM SEEK TIME MEASUREMENT
1074
1075                    THIS TEST MEASURES THE TIME REQUIRED TO SEEK FROM CYL  0  TO  CYL  632
1076                    OCT.  (#410 DEC.).  THIS TIME REPRESENTS THE MAXIMUM SEEK TIME, AND IT
1077                    IS MEASURED 128 TIMES IN EACH DIRECTION.  THE SPECIFIED  MAXIMUM  SEEK
1078                    TIME IS < 75 MILLI-SEC.
1079
1080                    SAMPLE PRINTOUT --
1081
1082                         MAXIMUM SEEK TIMES :
1083                         **FORWARD DIRECTION**
1084                         MIN = 68950 US
1085                         MAX = 69286 US  0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1086                         AVG = 69122 US
1087                         **REVERSE DIRECTION**
1088                         MIN = 70194 US
1089                         MAX = 70667 US  0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1090                         AVG = 70407 US
1091
1092
1093
1094                    9.3  TEST 17 - SECTOR ADDRESSING TEST
```

IN THIS TEST, ALL SECTORS OF FC, FT ARE WRITTEN, EACH WITH 256(DEC) WORDS OF ITS OWN SECTOR NO. + 100(OCT). NEXT, FOR EACH SECTOR WRITTEN A WRITE CHECK IS DONE, FOLLOWED BY A READ AND SOFTWARE COMPARE OF THE DATA.

9.4 TEST 20 - TRACK ADDRESSING TEST

IN THIS TEST, SECTOR FS OF CYL FC IS WRITTEN WITH 256 (DEC) WORDS OF THE TRACK NO. + 100(OCT). FOR EACH OF TRACKS 0,1,2, THEN, A WRITE CHECK OF EACH SECTOR IS DONE TO VERIFY THE WRITES. THEN, EACH OF THE 3 SECTORS IS RE-WRITTEN, AND AFTER EACH WRITE ALL OF THE THREE SECTORS ARE WRITE-CHECKED TO DETECT TRACK ADDRESSING PROBLEMS.

9.5 TEST 21 - READ/WRITE DATA TEST

THE READ/WRITE DATA TEST HAS 2 DIFFERENT VERSIONS, DEPENDING ON THE VALUE OF THE DATA PATTERN PARAMETER (PT).

FOR PT = 0 :

THIS TEST IS THE QUICK VERIFY DEFAULT DATA TEST, WHICH IS RUN WHEN PT=0, DUE TO PARAMETER INPUT CHOICE, OR ADDRESS 200 START. IN THIS TEST, THE ENTIRE PACK IS TESTED WITH A SINGLE DATA PATTERN. THIS PATTERN IS COMPRISED OF THE DATA PATTERNS 00-15, WHICH REPEATS EVERY 256(DEC) WORDS. EACH TRACK IS WRITTEN IN 4 SEGMENTS, FOLLOWED BY A WRITE-CHECK, READ, AND SOFTWARE COMPARE. THE SEGMENTS ARE 6 SECTORS EACH, WHICH MEANS THAT TRACK SPIRALING OCCURS ON THE LAST SEGMENT WRITTEN ON EACH TRACK.

FOR PT NOT = 0 :

THIS TEST PERFORMS READ/WRITE FUNCTIONS ON THE ENTIRE RANGE OF CYLINDERS (FC-LC), TRACKS (FT-LT), AND SECTORS (FS-LS) SPECIFIED. AT EACH SPECIFIED SECTOR WC WORDS OF THE CURRENT REPEATING DATA PATTERN ARE WRITTEN, AND THEN WRITE-CHECKED, FOLLOWED BY A READ AND SOFTWARE COMPARE. THIS IS DONE FOR ALL THE SPECFIEID SECTORS ON ALL THE TRACKS USING THE SECTOR INCREMENT IS, AND TRACK INCREMENT IT. AND THEN IT IS REPEATED USING EACH OF THE OTHER DATA PATTERNS CHOSEN IN PARAMETER PT, THEN, EACH OF THE ABOVE OPERATIONS ARE REPEATED AT EACH OF THE REMAINING CYLINDERS, IN THE SPECIFIED RANGE, USING THE CYLINDER INCREMENT IC. NOTE THAT FS MUST BE CHOSEN < OR = LS, AND FT MUST BE < OR = LT. HOWEVER, FC MAY BE <, =, OR > LC, AND IF FC>LC, THE CYLINDER ADDRESS WILL BE DECREMENTED BY IC (INSTEAD OF INCREMENTED) TO OBTAIN EACH NEW CYLINDER ADDRESS.

NOTE:

THE PACK ADDRESS LIMITS

(FT,LT,FS,LS,FC,LC) REFER TO THE RANGE OF DATA SECTORS AT WHICH TRANSFERS MAY BEGIN, USING THE SPECIFIED WORD COUNT WC. IF WC IS LARGE ENOUGH, HOWEVER, THE TRANSFERS MAY EXTEND BEYOND THE ABOVE LIMITS.

9.5.1 DATA PATTERNS

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 PRECOMPENSATI 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- RANDOM	15 USER- DEFINED
000001	072307		
000003	135143		
000007	156461		
000017	167230		
000037	073514		
000077	035646		
000177	016723		
000377	107351		
000777	143564		
001777	061672		
003777	030735		
007777	114356		
017777	046167		
037777	123073		
077777	151453		
177777	164616		

9.6 TEST 22 - DUAL-ACCESS DATA TEST

THIS TEST IS DESIGNED TO RUN ON 2 DIFFERENT PROCESSORS, SIMULTANEOUSLY AND INDEPENDENTLY PERFORMING DYNAMIC DATA OPERATIONS ON THE SAME DRIVE, THROUGH 2 DIFFERENT CONTROLLERS. TEST 22 HAS A SEPARATE STARTING ADDRESS = 220 AND IT IS NEVER RUN WITH THE OTHER TESTS IN AN AUTOMATIC MANNER. ALSO, BOTH PORTS MUST BE SWITCHED ON-LINE, AT THE DRIVE.

THIS TEST IS A MODIFIED VERSION OF READ/WRITE DATA TEST 21, IN WHICH A DRIVE RELEASE COMMAND IS DONE IMMEDIATELY AFTER COMPLETION OF READ WRITE OPERATIONS AT A PARTICULAR PACK ADDRESS. IN ALL OTHER RESPECTS, TESTS 21 AND 22 ARE IDENTICAL, AS FAR AS EACH PROCESSOR IS CONCERNED. THIS MEANS THAT THE TEST PARAMETERS (ADDRESS LIMITS, DATA PATTERNS, ETC.) CAN BE DEFINED DIFFERENTLY FOR EACH PROCESSOR. IF DIFFERENT DATA PATTERNS ARE USED, FOR EXAMPLE, THE DATA WHICH APPEARS ON THE DISK AT TIME OF FAILURE CAN IDENTIFY THE CONTROLLER FROM WHICH IT ORIGINATED. LIKEWISE, THE WORD COUNT MAY BE CHOSEN APPROPRIATELY SMALL (WC=1) TO CAUSE THE GREATEST AMOUNT OF CONTENTION FOR THE USE OF THE DRIVE.

NOTE THAT IF BOTH PROCESSORS ARE PERFORMING TRANSFERS WITH DIFFERENT DATA, AT THE SAME PACK ADDRESS, INTERFERENCE MAY OCCUR, IF ONE CONTROLLER SHOULD LOSE THE DRIVE PREMATURELY TO REPORT AN ERROR.

WHEN THE TEST IS STARTED (AT ADDRESS 220) PARAMETER INPUT MODE IS ENTERED, AND OPERATIONAL PARAMETERS AND THE DRIVE NUMBER MAY BE TYPED IN (SEE SECTION 8.1) AS WELL AS RK06 UNIBUS ADDRESS AND INTERRUPT VECTOR. THE TEST NUMBER IS AUTOMATICALLY SET TO 22, AND THE PROGRAM TYPES "ENTER T,O,S, OR R" AS DESCRIBED IN SECTION 8.2.3. AT THIS POINT, THE OPERATION OF THE PROGRAM IS THE SAME AS PREVIOUSLY DESCRIBED IN CONJUNCTION WITH THE OTHER TESTS. HOWEVER, ONLY TEST 22 MAY BE RUN, AND ONLY ON THE SINGLE SELECTED DRIVE. ALSO, THE OPERATOR MUST INDEPENDENTLY LOAD AND START THE TEST IN EACH PROCESSOR AND SELECT INPUT PARAMETERS, INDEPENDENTLY. THE DRIVE NUMBER TYPED AT EACH PROCESSOR CONSOLE MUST BE THE SAME, IN ORDER TO PERFORM THE TESTS ON THE SAME DRIVE, SIMULTANEOUSLY.

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

```
1326
1327
1328
1329                    4.  UNIT FIELD ERROR
1330                    5.  SUBSYSTEM TIMEOUT
1331                    6.  SERCON PARITY ERROR
1332                    7.  DRIVE DETECTED PARITY ERROR
1333                    8.  AC LOW
1334                    9.  SPEED LOSS
1335                   10.  ILLEGAL FUNCTION ERROR
1336                   11.  PROGRAMMING ERROR
1337                   12.  NON-EXECUTABLE FUNCTION ERROR
1338                   13.  DRIVE TYPE ERROR
1339                   14.  FORMAT ERROR
1340                   15.  WRITE LOCK ERROR
1341                   16.  DRIVE UNSAFE ERROR
1342                   17.  SEEK INCOMPLETE ERROR
1343                   18.  CYLINDER OVERFLOW ERROR
1344                   19.  ILLEGAL CYLINDER ADDRESS ERROR
1345                   20.  DRIVE OFF TRACK
1346                   21.  DRIVE TIMING ERROR
1347                   22.  DATA LATE ERROR
1348                   23.  CONTROLLER TIMEOUT ERROR
1349                   24.  OPERATION INCOMPLETE ERROR
1350                   25.  HEADER VRC ERROR
1351                   26.  DATA CHECK ERROR
1352                   27.  WRITE CHECK ERROR
1353                   28.  DATA MISCOMPARE
1354                   29.  NO DRIVE RESPONSE - UFE AND NXD
1355                   30.  DRIVE ERROR WILL NOT CLEAR
1356                   31.  DRIVE STATUS CHANGE WILL NOT CLEAR
1357                   32.  ATTENTION BUT NO STATUS CHANGE OR FAULT
1358                   33.  ATTENTION BUT DRIVE NOT AVAILABLE
1359                   34.  ERROR WHILE GATHERING DRIVE STATUS
1360                   35.  MULTIPLE DRIVE SELECT
1361                   36.  HEADER COMPARE ERROR
1362                   37.  ERROR IN RECALIBRATE FOR RECOVERY
1363                   38.  CLEAR CONTROLLER DID NOT CLEAR ERROR
1364                   39.  NO ATTENTION IN ATTENTION SUMMARY REGISTER
1365                   40.  UNSOLICITED ATTENTION
1366                   41.  UNEXPECTED DATA TYPE ERROR
1367                   42.  ATTENTION DID NOT RESET WITH CLEAR
1368                   43.  SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1369                   44.  DATA LATE WHEN UNLOADING HEADER
1370                   45.  CONTROLLER ERROR WHEN DRIVER SERVICING
1371                   46.  RETRY UNSUCCESSFUL
1372                   47.  BAD SECTOR ERROR ON SECTOR NOT LISTED BAD
1373
1374
1375
1376           10.2  ERROR HANDLING
1377
1378           ERRORS REPORTED BY THE PROGRAM CONSIST OF  COMMON  FAILURES  RESULTING
1379           FROM  ATTEMPTED  SUBSYSTEM FUNCTIONS. AS WELL AS CERTAIN ERRORS UNIQUE
1380           TO PARTICULAR  TESTS.   EACH  ERROR  PRINTOUT  CONSISTS  OF  AN  ERROR
1381           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 ON A GIVEN ERROR. IN THE CASE OF DATA MISCOMPARE ERRORS, FOR INSTANCE, THE WORD 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.

MANY OF THE COMMON ERRORS ARE RETRIED SEVERAL TIMES, AND IF THEY PERSIST, AN ABORT IS MADE. DATA ERRORS AND MISCOMPARES ARE REPORTED, AND THE PROGRAM CONTINUES TESTING.

10.3 ERROR PRINTOUT EXAMPLES

EXAMPLE 1:

```
** NON-EXISTANT DRIVE ERROR
   TEST 2

PREVIOUS COMMAND :
DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
000000  000177  000000  000000  000000  000000
  HI BA   LO BA
  000000  000000

CURRENT COMMAND :
ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
037576  000000  000125  000000  000000  000000  000000
  HI BA   LO BA
  000000  000000

RK611 REGISTERS:
RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF
140324  010100  000000  000000  000000  000000  000400

REMAINING REGISTERS NOT VALID
```

EXAMPLE 2:

```
**  DATA MISCOMPARE
    TEST 21

PREVIOUS COMMAND :
DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
000000  000121  000000  000000  000000  036000
  HI BA   LO BA
  000000  074000

CURRENT COMMAND :
ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT
```

034606 000000 000121 000000 000000 000000 036000
HI BA LO BA
000000 074000

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

MC #	GOOD	BAD	HI PHY	LO PHY	VRT AD	KIPAR6
000000	000001	125252	000000	074000	154000	000600
000001	000002	125252	000000	074002	154002	000600
000002	000003	125252	000000	074004	154004	000600
000003	000004	125252	000000	074006	154006	000600
000004	000005	125252	000000	074010	154010	000600
000005	000006	125252	000000	074012	154012	000600
000006	000007	125252	000000	074014	154014	000600
000007	000010	125252	000000	074016	154016	000600
000010	000011	125252	000000	074020	154020	000600
000011	000012	125252	000000	074022	154022	000600

APPENDIX A

SAMPLE ADDRESS 200 DEFAULT RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART 1, 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.

```
     DZR6M-C - RK611 RK06 SUBSYSTEM VERIFICATION : PART 1

     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

     ROTATIONAL TIMES :
     MIN = 24153 US   99 OF 128 BELOW SPEC'S MIN OF 24375 US
     MAX = 25125 US    0 OF 128 ABOVE SPEC'D MAX OF 25625 US
     AVG = 24374 US

     ONE CYL SEEK TIMES :
     **FORWARD DIRECTION**
     MIN = 6323 US
     MAX = 6686 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
     AVG = 6518 US
     **REVERSE DIRECTION**
     MIN = 6332 US
     MAX = 6713 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
     AVG = 6513 US

     AVERAGE SEEK TIMES :
```

```
                                        **FORWARD DIRECTION**
                                        AVG = 35673 US   SPEC'D MAX IS 38000 US
                                        **REVERSE DIRECTION**
                                        AVG = 35823 US   SPEC'D MAX IS 38000 US

                                        MAXIMUM SEEK TIMES :
                                        **FORWARD DIRECTION**
                                        MIN = 68950 US
                                        MAX = 69150 US   0 OF 128 ABOVE SPEC'D MAX OF 75000 US
                                        AVG = 69064 US
                                        **REVERSE DIRECTION**
                                        MIN = 70222 US
                                        MAX = 70530 US   0 OF 128 ABOVE SPEC'D MAX OF 75000 US
                                        AVG = 70379 US

                                        END PASS #    1
```

APPENDIX B

SAMPLE ADDRESS 204 RUN

THE FOLLOWING PRINTOUT WAS THE RESULT OF RUNNING SUBSYSTEM VERIFICATION, PART 1. 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 2 BE RUN 25(OCTAL) TIMES, TEST 7 FOUR TIMES, TEST 14 RUN 2 TIMES, AND TEST 21 ONCE. THESE TESTS WERE SPECIFIED TO RUN FOR 2 PROGRAM PASSES, USING PARAMETER VALUES TYPED BY THE OPERATOR. IN THIS CASE PACK ADDRESS PARAMETERS AND DATA TRANSFER WORD COUNT WERE SPECIFIED, FOR TESTS WHICH UTILIZE THEM.

```
DZR6M-C - RK611,RK06 SUBSYSTEM VERIFICATION : PART 1

LAST PHYS MEM ADR = 377776


PARAMETER INPUT MODE

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

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
*

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

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

```
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
```

```
*

TEST   ITERATIONS
  1          10 * 0
  2         200 * 25
  3          10 * 0
  4          10 * 0
  5           1 * 0
  6           1 * 0
  7           1 * 4
 10          40 * 0
 11         500 * 0
 12           2 * 0
 13           1 * 0
 14           1 * 2
 15           1 * 0
 16           1 * 0
 17          10 * 0
 20          10 * 0
 21           1 *

ENTER L, C, OR I
* L

TEST   ITERATIONS
  1           0
  2          25
  3           0
  4           0
  5           0
  6           0
  7           4
 10           0
 11           0
 12           0
 13           0
 14           2
 15           0
 16           0
 17           0
 20           0
 21           1

ENTER L, C, OR I
* I

T = TYPE PARAMETER LIST
O = OPEN PARAMETER LIST
S = SET INDIVIDUAL PARAM
R = RUN TESTS

ENTER T, O, S, OR R
* O
```

```
1648
1649
1650
1651                                    TO DEFAULT ALL PARAMETERS TYPE D<CR>, ELSE <CR>
1652                                    *
1653                                    FC=0 *
1654                                    LC=632 * 2
1655                                    IC=1 *
1656                                    FT=0 * 1
1657                                    LT=2 * 1
1658                                    IT=1 *
1659                                    S0=0 *
1660                                    S1=23 *
1661                                    S2=0 * 10
1662                                    S3=25 * 10
1663                                    IS=1 *
1664                                    PT=0 * 20
1665                                    MA=61566 *
1666                                    WC=403 * 1100
1667                                    CS=0 *
1668                                    ST=0 *
1669                                    SM=1000 *
1670
1671                                    ENTER T, O, S, OR R
1672                                    * T
1673                                    FC=0
1674                                    LC=2
1675                                    IC=1
1676                                    FT=1
1677                                    LT=1
1678                                    IT=1
1679                                    S0=0
1680                                    S1=23
1681                                    S2=10
1682                                    S3=10
1683                                    IS=1
1684                                    PT=20
1685                                    MA=61566
1686                                    WC=1100
1687                                    CS=0
1688                                    ST=0
1689                                    SM=1000
1690
1691                                    ENTER T, O, S, OR R
1692                                    * R
1693                                    ENTER NO. OF PASSES (1-77777) :
1694                                    * 2
1695
1696                                    TESTING DRIVE 0
1697                                    DRIVE SER. NO.  8
1698                                    CART. SER. NO.  334
1699
1700                                    ONE CYL SEEK TIMES :
1701                                    **FORWARD DIRECTION**
1702                                    MIN = 6319 US
1703                                    MAX = 6691 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
```

```
1704
1705
1706
1707                    AVG = 6502 US
1708                    **REVERSE DIRECTION**
1709                    MIN = 6301 US
1710                    MAX = 6682 US    0 OF 410 ABOVE SPEC'D MAX OF 9000 US
1711                    AVG = 6498 US
1712
1713                    ONE CYL SEEK TIMES :
1714                    **FORWARD DIRECTION**
1715                    MIN = 6332 US
1716                    MAX = 6695 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1717                    AVG = 6506 US
1718                    **REVERSE DIRECTION**
1719                    MIN = 6314 US
1720                    MAX = 6686 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1721                    AVG = 6503 US
1722
1723                    ONE CYL SEEK TIMES :
1724                    **FORWARD DIRECTION**
1725                    MIN = 6319 US
1726                    MAX = 6691 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1727                    AVG = 6504 US
1728                    **REVERSE DIRECTION**
1729                    MIN = 6301 US
1730                    MAX = 6700 US    0 OF 410 ABOVE SPEC'D MAX OF 3000 US
1731                    AVG = 6500 US
1732
1733                    ONE CYL SEEK TIMES :
1734                    **FORWARD DIRECTION**
1735                    MIN = 6326 US
1736                    MAX = 6698 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1737                    AVG = 6511 US
1738                    **REVERSE DIRECTION**
1739                    MIN = 6317 US
1740                    MAX = 6698 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1741                    AVG = 6507 US
1742
1743                    END PASS #    2
1744                    TO TEST ALL DRIVES TYPE "A"<CR>, ELSE <CR>
1745                    *
1746
1747
1748
1749
```

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


                                .NLIST  MC,MD,CND
                                .LIST   ME
                                .ENABL  ABS,AMA
167000                          $SWR=   167000
                        ;;*****************************************************************
                        .SBTTL  STARTING ADDRESSES
                        ;*
                        ;*      200     DEFAULT PARAMETERS FOR TESTS 1-21
                        ;*      204     SELECT PARAMETERS FOR TESTS 1-21
                        ;*      220     DUAL-ACCESS DATA TEST
                        ;;*****************************************************************

000001                  $TN=1
                        .TITLE  MD-11-DZR6M-C - RK611/RK06 SUBSYS. VERIF. : PART 1
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DAVE HOFFMAN
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            REPORT DESCRIPTION ONLY, ON ERRORS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            APPLY RANDOM STALL BETWEEN OPERATIONS
                        ;*         7            DO EXPLICIT SEEKS IN TESTS 1-12
                        ;*         6            REPORT 1 ERROR PER TRANSFER IN TESTS 17,21
                        ;*         5            INHIBIT WRITES IN TEST 21
                        ;*         4            INHIBIT WRITE CHECKS IN TEST 21
                        ;*         3            INHIBIT READS AND SOFTWARE COMPARES IN TEST 21
                        ;*         2            INHIBIT SOFTWARE COMPARES IN TEST 21
                        ;*         1            READ AFTER A WRITE CHECK ERROR IN TEST 21
                        ;*         0            REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 17,21
                        .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
                        ;*        03            INHIBIT TIMING REPORTS IN TESTS 13-16
```

```
                        .SBTTL  BASIC DEFINITIONS

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

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

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

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

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

```
                    000002          SWC1=   2
                    000001          SWC0=   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
                                
 177572                         SR0=    177572
 177574                         SR1=    177574
 177576                         SR2=    177576
 172516                         SR3=    172516
                                
                                ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
                                
 172300                         KIPDR0= 172300
 172302                         KIPDR1= 172302
 172304                         KIPDR2= 172304
 172306                         KIPDR3= 172306
 172310                         KIPDR4= 172310
 172312                         KIPDR5= 172312
 172314                         KIPDR6= 172314
 172316                         KIPDR7= 172316
                                
                                ;*KERNEL "I" PAGE ADDRESS REGISTERS
                                
 172340                         KIPAR0= 172340
 172342                         KIPAR1= 172342
 172344                         KIPAR2= 172344
 172346                         KIPAR3= 172346
 172350                         KIPAR4= 172350
 172352                         KIPAR5= 172352
 172354                         KIPAR6= 172354
 172356                         KIPAR7= 172356
                                
 170200                         MAPLO0=170200
 170202                         MAPHO0=170202
 172100                         MEMCSR=172100                   ;MEMORY CSR REG START ADRS
 177740                         LOERAD=177740                   ;11/70 MEM LO ERROR ADRS REG
 177742                         HIERAD=177742                   ;11/70 MEM HI ERROR ADRS REG
 177744                         MEMSYS=177744                   ;11/70 MEM SYSTEM REG
                                .SBTTL  TRAP CATCHER
                                
 000000                         .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
 000174                         .=174
 000174  000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
 000176  000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
 000200  000137  012456                JMP     @#DFSTRT        ;;JUMP TO STARTING ADDRESS OF PROGRAM
 000166                         .=166
 000166  000000                HZ:     .WORD   0               ;LINE FREQ. FLAG -  0 = 60 HZ
 000204                         .=204
 000204  000137  012530                JMP     @#PSTART
 000220                         .=220
 000220  000137  012512                JMP     @#DASTRT
```

```
1974  000224  000700                  ..LOW:  .WORD   700
1975  000226  001100                  ..HIGH: .WORD   1100
1976                                  .SBTTL  ACT11 HOOKS
1977
1978                                  ;;******************************************************************
1979                                  ;HOOKS REQUIRED BY ACT11
1980          000230                          $SVPC=.                 ;SAVE PC
1981          000046                          .=46
1982  000046  025056                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$ECP
1983          000052                          .=52
1984  000052  140000                          .WORD   140000          ;;2)SET LOC.52 TO 140000
1985          000230                          .=$SVPC                 ;; RESTORE PC
1986          001000                          .=1000
1987                                  .SBTTL  APT PARAMETER BLOCK
1988
1989                                  ;;******************************************************************
1990                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1991                                  ;;******************************************************************
1992          001000                          .$X=.   ;;SAVE CURRENT LOCATION
1993          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1994  000024  000200                          200     ;;FOR APT START UP
1995          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
1996  000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
1997          001000                          .=.$X   ;;RESET LOCATION COUNTER
1998                                  ;;******************************************************************
1999                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
2000                                  ;INTERFACE SPEC.
2001
2002  001000                          $APTHD:
2003  001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
2004  001002  001320                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
2005  001004  001320                  $TSTM:  .WORD   1320    ;;RUN TIM OF LONGEST TEST
2006  001006  001546                  $PASTM: .WORD   1546    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
2007  001010  001546                  $UNITM: .WORD   1546    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
2008  001012  000030                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
2009          120210                  AVECT1=120210                   ;RKVEC=210, RKPRI=5
2010          177440                  ABASE=177440                    ;RKBAS ADRS
2011          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)
```

```
001224  000000          $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
001226  000000          $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
001230  000000          $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
001232  000000          $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
001234  000000          $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
001236  000000          $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
001240  000000          $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
001242  000000          $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
001244  000000          $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
001246  000000          $REG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
001250  000000          $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
001252  000000          $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
001254  000000          $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
001256  000000          $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
001260  000000          $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
001262  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001264  000000          $TMP1:  .WORD   0               ;;USER DEFINED
001266  000000          $TMP2:  .WORD   0               ;;USER DEFINED
001270  000000          $TMP3:  .WORD   0               ;;USER DEFINED
001272  000000          $TMP4:  .WORD   0               ;;USER DEFINED
001274  000000          $TMP5:  .WORD   0               ;;USER DEFINED
001276  000000          $TMP6:  .WORD   0               ;;USER DEFINED
001300  000000          $TMP7:  .WORD   0               ;;USER DEFINED
001302  000000          $TMP10: .WORD   0               ;;USER DEFINED
001304  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001306  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001310  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001314     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001315     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001316  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;**************************************************************
                        .SBTTL  APT MAILBOX-ETABLE

                        ;;**************************************************************
                        .EVEN
001320                  $MAIL:                          ;;APT MAILBOX
001320  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001322  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001324  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001326  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
001330  000000          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001332  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001334  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001336  000000          $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001340                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
001340     000          $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001341     000          $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001342  000000          $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001344  000000          $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001346  000000          $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                        ;*                              BITS 15-11=CPU TYPE
                        ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                                      11/70=06,PDQ=07,Q=10
                        ;*                              BIT 10=REAL TIME CLOCK
                        ;*                              BIT  9=FLOATING POINT PROCESSOR
                        ;*                              BIT  8=MEMORY MANAGEMENT
```

```
001350        000          $MAMS1: .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001351        000          $MTYP1: .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
                           ;*                       MEM.TYPE BYTE   --  (HIGH BYTE)
                           ;*                                900 NSEC CORE=001
                           ;*                                300 NSEC BIPOLAR=002
                           ;*                                500 NSEC MOS=003
001352     000000          $MADR1: .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                           ;*                         MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001354        000          $MAMS2: .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001355        000          $MTYP2: .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001356     000000          $MADR2: .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001360        000          $MAMS3: .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001361        000          $MTYP3: .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001362     000000          $MADR3: .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001364        000          $MAMS4: .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001365        000          $MTYP4: .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001366     000000          $MADR4: .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001370     120210          $VECT1: .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001372     000000          $VECT2: .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001374     177440          $BASE:  .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001376     000377          $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
001400                     $ETEND:
                           .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  055577                         EM1
 001402  060672                         DH100
 001404  062456                         DT100
 001406  062572                         DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
 001410  055623                         EM2
 001412  060672                         DH100
 001414  062456                         DT100
 001416  062572                         DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
 001420  055647                         EM3
 001422  060672                         DH100
 001424  062456                         DT100
 001426  062572                         DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
 001430  055672                         EM4
 001432  060672                         DH100
 001434  062456                         DT100
 001436  062572                         DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
 001440  055713                         EM5
 001442  060672                         DH100
 001444  062456                         DT100
 001446  062622                         DF02

                                        ;ERROR 6
 001450  055732                         EM6                     ;D TO C PARITY ERROR
 001452  060672                         DH100
 001454  062456                         DT100
 001456  062656                         DF03

                                        ;ERROR 7
 001460  055756                         EM7                     ;DRIVE DETECTED PARITY ERROR
 001462  060672                         DH100
 001464  062456                         DT100
```

```
                001466  062656                  DF03
                                                ;ERROR 10
                001470  056012                  EM10            ;AC LOW
                001472  060672                  DH100
                001474  062456                  DT100
                001476  062622                  DF02
                                                ;ERROR 11
                001500  056021                  EM11            ;SPEED LOSS
                001502  060672                  DH100
                001504  062456                  DT100
                001506  062622                  DF02
                                                ;ERROR 12
                001510  056034                  EM12            ;ILLEGAL FUNCTION
                001512  060672                  DH100
                001514  062456                  DT100
                001516  062622                  DF02
                                                ;ERROR 13
                001520  056055                  EM13            ;PROGRAMMING ERROR
                001522  060672                  DH100
                001524  062456                  DT100
                001526  062572                  DF01
                                                ;ERROR 14       ;NON-EXISTANT FUNCTION
                001530  056077                  EM14
                001532  060672                  DH100
                001534  062456                  DT100
                001536  062622                  DF02
                                                ;ERROR 15
                001540  056125                  EM15            ;DRIVE TYPE ERROR
                001542  060672                  DH100
                001544  062456                  DT100
                001546  062622                  DF02
                                                ;ERROR 16
                001550  056146                  EM16            ;FORMAT ERROR
                001552  060672                  DH100
                001554  062456                  DT100
                001556  062622                  DF02
                                                ;ERROR 17
                001560  056163                  EM17            ;WRITE LOCK ERROR
                001562  060672                  DH100
                001564  062456                  DT100
                001566  062622                  DF02
                                                ;ERROR 20
                001570  056204                  EM20            ;DRIVE UNSAFE
                001572  060672                  DH100
                001574  062456                  DT100
                001576  062622                  DF02
```

```
2259                                    ;ERROR 21
2260  001600  056221                    EM21                    ;SEEK INCOMPLETE
2261  001602  060672                    DH100
2262  001604  062456                    DT100
2263  001606  062622                    DF02
2264
2265                                    ;ERROR 22
2266  001610  056241                    EM22                    ;CYLINDER OVERFLOW
2267  001612  060672                    DH100
2268  001614  062456                    DT100
2269  001616  062622                    DF02
2270
2271                                    ;ERROR 23
2272  001620  056263                    EM23                    ;ILLEGAL CYLINDER
2273  001622  060672                    DH100
2274  001624  062456                    DT100
2275  001626  062622                    DF02
2276
2277                                    ;ERROR 24
2278  001630  056314                    EM24                    ;DRIVE OFF TRACK
2279  001632  060672                    DH100
2280  001634  062456                    DT100
2281  001636  062622                    DF02
2282
2283                                    ;ERROR 25
2284  001640  056334                    EM25                    ;DRIVE TIMING ERROR
2285  001642  060672                    DH100
2286  001644  062456                    DT100
2287  001646  062622                    DF02
2288
2289                                    ;ERROR 26
2290  001650  056357                    EM26                    ;DATA LATE
2291  001652  060672                    DH100
2292  001654  062456                    DT100
2293  001656  062622                    DF02
2294
2295                                    ;ERROR 27
2296  001660  056371                    EM27                    ;CONTROLLER TIMEOUT
2297  001662  060672                    DH100
2298  001664  062456                    DT100
2299  001666  062622                    DF02
2300
2301                                    ;ERROR 30               ;OPERATION INCOMPLETE
2302  001670  056414                    EM30
2303  001672  060672                    DH100
2304  001674  062456                    DT100
2305  001676  062732                    DF05
2306
2307                                    ;ERROR 31
2308  001700  056441                    EM31                    ;HEADER VRC ERROR
2309  001702  060672                    DH100
2310  001704  062456                    DT100
2311  001706  062732                    DF05
2312
2313                                    ;ERROR 32
2314  001710  056462                    EM32                    ;DATA CHECK ERROR
```

```
2315  001712  060672                  DH100
2316  001714  062456                  DT100
2317  001716  062766                  DF07
2318
2319                                  ;ERROR 33
2320  001720  056503                  EM33          ;WRITE CHECK ERROR
2321  001722  060672                  DH100
2322  001724  062456                  DT100
2323  001726  063022                  DF10
2324
2325                                  ;ERROR 34
2326  001730  056525                  EM34          ;DATA MISCOMPARE(S)
2327  001732  060672                  DH100
2328  001734  062456                  DT100
2329  001736  062716                  DF04
2330
2331                                  ;ERROR 35
2332  001740  056545                  EM35          ;NO DRIVE RESPONSE-UFE & NXD
2333  001742  060672                  DH100
2334  001744  062456                  DT100
2335  001746  062572                  DF01
2336
2337                                  ;ERROR 36
2338  001750  056601                  EM36          ;DRIVE ERROR WILL NOT CLEAR
2339  001752  000000                  0
2340  001754  000000                  0
2341  001756  000000                  0
2342
2343                                  ;ERROR 37     ;DRIVE STATUS CHANGE WILL NOT CLEAR
2344  001760  056634                  EM37
2345  001762  000000                  0
2346  001764  000000                  0
2347  001766  000000                  0
2348
2349                                  ;ERROR 40
2350  001770  056677                  EM40          ;ATTENTION BUT NO STATUS CHANGE OR FAULT
2351  001772  060672                  DH100
2352  001774  062456                  DT100
2353  001776  062622                  DF02
2354
2355                                  ;ERROR 41
2356  002000  056743                  EM41          ;ATTENTION BUT DRIVE NOT AVAILABLE
2357  002002  060672                  DH100
2358  002004  062456                  DT100
2359  002006  062622                  DF02
2360
2361                                  ;ERROR 42
2362  002010  057001                  EM42          ;ATTENTION WHEN NOT EXPECTED
2363  002012  060672                  DH100
2364  002014  062456                  DT100
2365  002016  062622                  DF02
2366
2367                                  ;ERROR 43     ;ERROR WHILE GATHERING DRIVE STATUS
2368  002020  057031                  EM43
2369  002022  060672                  DH100
2370  002024  062456                  DT100
```

```
2371  002026  063066                    DF12
2372
2373                                    ;ERROR 44
2374  002030  057306                    EM63        ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2375  002032  060672                    DH100
2376  002034  062456                    DT100
2377  002036  063066                    DF12
2378
2379                                    ;ERROR 45
2380  002040  057353                    EM64        ;NO ATTENTION IN ATTENTION SUMMARY REG
2381  002042  060672                    DH100
2382  002044  062456                    DT100
2383  002046  063066                    DF12
2384
2385                                    ;ERROR 46
2386  002050  057416                    EM65        ;UNSOLICITED ATTENTION
2387  002052  060672                    DH100
2388  002054  062456                    DT100
2389  002056  063066                    DF12
2390
2391                                    ;ERROR 47
2392  002060  057444                    EM66        ;UNEXPECTED DATA TYPE ERROR
2393  002062  060672                    DH100
2394  002064  062456                    DT100
2395  002066  063066                    DF12
2396
2397                                    ;ERROR 50
2398  002070  057477                    EM67        ;ATTENTION DID NOT RESET WITH CLEAR
2399  002072  060672                    DH100
2400  002074  062456                    DT100
2401  002076  063066                    DF12
2402
2403                                    ;ERROR 51
2404  002100  057536                    EM70        ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2405  002102  060672                    DH100
2406  002104  062456                    DT100
2407  002106  063066                    DF12
2408
2409                                    ;ERROR 52
2410  002110  057066                    EM52        ;MULTIPLE DRIVE SELECT
2411  002112  060672                    DH100
2412  002114  062456                    DT100
2413  002116  063066                    DF12
2414
2415                                    ;ERROR 53
2416  002120  057114                    EM53        ;ABREVIATED HCE ERROR
2417  002122  060672                    DH100
2418  002124  062456                    DT100
2419  002126  063116                    DF13
2420
2421                                    ;ERROR 54
2422  002130  056414                    EM30        ;OPERATION INCOMPLETE ERROR
2423  002132  060672                    DH100
2424  002134  062456                    DT100
2425  002136  063146                    DF14
2426
```

```
                                        ;ERROR 55
002140  056441                          EM31                    ;ABREVIATED HVRC ERROR
002142  060672                          DH100
002144  062456                          DT100
002146  063116                          DF13

                                        ;ERROR 56
002150  057141                          EM56                    ;2 TIMEOUT ERROR
002152  060672                          DH100
002154  062456                          DT100
002156  063176                          DF15

                                        ;ERROR 57               ;2ND LEVEL IN SUBSYSTEM TIMEOUT
002160  057141                          EM56
002162  060672                          DH100
002164  062456                          DT100
002166  063242                          DF16

                                        ;ERROR 60
002170  057160                          EM60                    ;ERROR IN RECAL FOR RECOVERY
002172  000000                          0
002174  000000                          0
002176  000000                          0

                                        ;ERROR 61
002200  057214                          EM61                    ;ABORT MESSAGE
002202  000000                          0
002204  000000                          0
002206  000000                          0

                                        ;ERROR 62
002210  057262                          EM62                    ;CYLINDER MISCOMPARE
002212  060672                          DH100
002214  062456                          DT100
002216  063306                          DF17

                                        ;ERROR 63               ;DATA ERROR WORDS
002220  000000                          0
002222  000000                          0
002224  062550                          DT602
002226  063416                          DF25

                                        ;ERROR 64
002230  057306                          EM63                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
002232  060672                          DH100
002234  062456                          DT100
002236  062622                          DF02

                                        ;ERROR 65
002240  057353                          EM64                    ;NO ATTENTION IN ATTENTION SUMMARY REG
002242  060672                          DH100
002244  062456                          DT100
002246  062622                          DF02

                                        ;ERROR 66
002250  057416                          EM65                    ;UNSOLICITED ATTENTION
```

```
2483  002252  060672                    DH100
2484  002254  062456                    DT100
2485  002256  062622                    DF02
2486
2487                                    ;ERROR 67
2488  002260  057444                    EM66        ;UNEXPECTED DATA TYPE ERROR
2489  002262  060672                    DH100
2490  002264  062456                    DT100
2491  002266  062622                    DF02
2492
2493                                    ;ERROR 70
2494  002270  057477                    EM67        ;ATTENTION DID NOT RESET WITH CLEAR
2495  002272  060672                    DH100
2496  002274  062456                    DT100
2497  002276  062622                    DF02
2498
2499                                    ;ERROR 71
2500  002300  057536                    EM70        ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
2501  002302  060672                    DH100
2502  002304  062456                    DT100
2503  002306  062622                    DF02
2504
2505                                    ;ERROR 72
2506  002310  057605                    EM71        ;DATA LATE WHEN UNLOADING HEADER
2507  002312  060672                    DH100
2508  002314  062456                    DT100
2509  002316  062622                    DF02
2510
2511                                    ;ERROR 73
2512  002320  057645                    EM72        ;CONTROLLER ERROR DURING DRIVER SERVICE
2513  002322  060672                    DH100
2514  002324  062456                    DT100
2515  002326  062622                    DF02
2516
2517                                    ;ERROR 74
2518  002330  057714                    EM73        ;DRIVE DETECTED PARITY ERROR
2519  002332  060672                    DH100
2520  002334  062456                    DT100
2521  002336  062622                    DF02
2522
2523                                    ;ERROR 75
2524  002340  057750                    EM74        ;UNDEFINED ERROR
2525  002342  060672                    DH100
2526  002344  062456                    DT100
2527  002346  062622                    DF02
2528
2529                                    ;ERROR 76
2530  002350  057770                    EM75        ;MARKING SECTOR BAD MESSAGE
2531  002352  000000                    0
2532  002354  000000                    0
2533  002356  000000                    0
2534
2535                                    ;ERROR 77
2536  002360  060023                    EM76        ;LAD DATA VERIFICATION WITH READ
2537  002362  061657                    DH605
2538  002364  062542                    DT601
```

```
2539  002366  063342                        DF21
2540
2541                                        ;ERROR 100
2542  002370  060102                        EM77            ;RETRY SUCCESFUL MESSAGE
2543  002372  000000                        0
2544  002374  062542                        DT601
2545  002376  063402                        DF23
2546
2547                                        ;ERROR 101
2548  002400  060102                        EM77            ;ANOTHER RETRY SUCCESSFUL MESSAGE
2549  002402  000000                        0
2550  002404  062542                        DT601
2551  002406  063402                        DF23
2552
2553                                        ;ERROR 102
2554  002410  060123                        EM100           ;RETRY UNSUCCESSFUL MESSAGE
2555  002412  000000                        0
2556  002414  062542                        DT601
2557  002416  063402                        DF23
2558
2559                                        ;ERROR 103
2560  002420  060146                        EM101           ;NO VALID HEADERS IN TRACK JUST READ
2561  002422  061641                        DH6042
2562  002424  062542                        DT601
2563  002426  063412                        DF24
2564
2565                                        ;ERROR 104
2566  002430  060224                        EM102           ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2567  002432  060672                        DH100
2568  002434  062456                        DT100
2569  002436  062622                        DF02
2570
2571                                        ;ERROR  105
2572  002440  060276                        EM103           ;TIMED-OUT ON READ HEADER
2573  002442  060672                        DH100
2574  002444  062456                        DT100
2575  002446  062656                        DF03
2576
2577                                        ;ERROR  106
2578  002450  060324                        EM104           ;TIMED-OUT ON SEEK
2579  002452  060672                        DH100
2580  002454  062456                        DT100
2581  002456  062656                        DF03
2582
2583                                        ;ERROR  107
2584  002460  060346                        EM105           ;DRIVE SIEZED BY OTHER PORT
2585  002462  060672                        DH100
2586  002464  062456                        DT100
2587  002466  062622                        DF02
2588
2589                                        ;ERROR  110
2590  002470  060401                        EM106           ;"DATA MISCMPR WHILE BAI SET"
2591  002472  060672                        DH100
2592  002474  062456                        DT100
2593  002476  063432                        DF27
2594
```

```
2595                                    ;ERROR  111
2596  002500  060434                   EM107                   ;"NO NEM WHEN EXPECTED"
2597  002502  000000                   0
2598  002504  000000                   0
2599  002506  000000                   0
2600
2601                                    ;ERROR  112
2602  002510  060501                   EM110                   ;"INTRPT WHEN CNTRLR NOT READY"
2603  002512  060672                   DH100
2604  002514  062456                   DT100
2605  002516  062572                   DF01
2606
2607                                    ;ERROR  113
2608  002520  060534                   EM111                   ;"NO ATT'N ON SEEK"
2609  002522  060672                   DH100
2610  002524  062456                   DT100
2611  002526  062622                   DF02
2612
2613                                    ;ERROR  114
2614  002530  060555                   EM112                   ;DRIVE'S CYLINDER INCORRECT
2615  002532  062066                   DH702
2616  002534  062516                   DT202
2617  002536  063422                   DF26
2618
2619                                    ;ERROR  115
2620  002540  000000                   0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2621  002542  000000                   0
2622  002544  062542                   DT601
2623  002546  063046                   DF11
2624
2625                                    ;ERROR  116
2626  002550  056525                   EM34                    ;DATA MISCOMPARE (11/70)
2627  002552  061203                   DH103
2628  002554  062566                   DT103
2629  002556  063332                   DF20
2630
2631                                    ;ERROR  117
2632  002560  000000                   0                       ;PART OF DATA MISCOMPARE
2633  002562  000000                   0
2634  002564  062456                   DT100
2635  002566  063352                   DF22
2636
2637                                    ;ERROR  120
2638  002570  060610                   EM113                   ;ABORT- CAN'T READ BSF
2639  002572  060672                   DH100
2640  002574  062456                   DT100
2641  002576  062572                   DF01
2642
2643                                    ;ERROR  121
2644  002600  060636                   EM114                   ;KT11 FAILURE
2645  002602  060672                   DH100
2646  002604  062456                   DT100
2647  002606  063456                   DF30
2648
2649                                    ;ERROR  122
2650  002610  060653                   EM115                   ;MEM PARITY ERROR
```

```
2651  002612  060672                    DH100
2652  002614  062456                    DT100
2653  002616  063502                    DF31
2654
2655
2656
2657
2658
2659
```

```
                        .SBTTL  BIT ASSIGNMENTS IN THE RK611 REGISTERS

                        ;RKCS1
                        ; 15      14      13      12      11      10      9       8
                        ;-------------------------------------------------------------
                        ;CERR  !  DI   !  DCPAR!  CFNT  !  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                             !
                        ;-------------------------------------------------------------
```

BIT ASSIGNMENTS IN THE Rk611 REGISTERS

RKDB (DATA BUFFER)

15	14	13	12	11	10	9	8
DB15	DB14	DB13	DB12	DB11	DB10	DB9	DB8
READ/WRITE							

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

RkASOF (ATTENTION SUMMARY AND OFFSET)

15	14	13	12	11	10	9	8
ATN7	ATN6	ATN5	ATN4	ATN3	ATN2	ATN1	ATN0
READ ONLY							

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

RKWC (WORD COUNT)

15	14	13	12	11	10	9	8
WC15	WC14	WC13	WC12	WC11	WC10	WC9	WC8
READ/WRITE							

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

RkBA (BUS ADDRESS)

15	14	13	12	11	10	9	8
BA15	BA14	BA13	BA12	BA11	BA10	BA9	BA8
READ/WRITE							

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

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

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

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

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

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

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

```
;RKECC PAT
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
;                     READ ONLY                       !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
;EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
;                     READ ONLY                       !
;-----------------------------------------------------

;RKECC/POS
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; UN  !  UN  !  UN  ! EPO12! EPO11! EPO10! EPO9 ! EPO8 !
;                     READ ONLY                       !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
;EP07 ! EP06 ! EP05 ! EP04 ! EP03 ! EP02 ! EP01 ! EP00 !
;                     READ ONLY                       !
;-----------------------------------------------------

;DRIVE STATUS INFORMATION

;LINE A MESSAGE 00
; 15     14     13     12     11     10     9      8
;-----------------------------------------------------
; PAR ! DSC  ! PIP  ! SPON ! WRLK ! OFFON! FMT  ! DRTP !
;-----------------------------------------------------

;  7      6      5      4      3      2      1      0
;-----------------------------------------------------
;DRDY !  VV  ! DRA  !  NU  !  NU  ! DRIVE SELECT CODE  !
;-----------------------------------------------------
```

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

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

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

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

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

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

```
                                ;LINE A MESSAGE 10
                                ; 15     14     13     12     11     10     9      8
                                ;-----------------------------------------------------
                                ; PAR !  NU  !  NU  ! CYLINDER DIFF/OFFSET VALUE      !
                                ;-----------------------------------------------------
                                ;
                                ;  7      6      5      4      3      2      1      0
                                ;-----------------------------------------------------
                                ;CYLINDER DIFF/OFFSET VALUE!  NU  !  DRIVE SELECT CODE !
                                ;-----------------------------------------------------
                                ;
                                ;LINE B MESSAGE 10
                                ; 15     14     13     12     11     10     9      8
                                ;-----------------------------------------------------
                                ; PAR ! ALIGN!  NU  !    CYLINDER ADDRESS             !
                                ;     ! SIGN !                                       !
                                ;-----------------------------------------------------
                                ;
                                ;  7      6      5      4      3      2      1      0
                                ;-----------------------------------------------------
                                ;    CYLINDER ADDRESS      ! UN ! UN !  1 !  0 !
                                ;-----------------------------------------------------
                                ;
                                ;LINE A MESSAGE 11
                                ; 15     14     13     12     11     10     9      8
                                ;-----------------------------------------------------
                                ; PAR !     DRIVE SERIAL NUMBER                      !
                                ;-----------------------------------------------------
                                ;
                                ;  7      6      5      4      3      2      1      0
                                ;-----------------------------------------------------
                                ; DRIVE SERIAL NUMBER             !  DRIVE SELECT CODE !
                                ;-----------------------------------------------------
                                ;
                                ;LINE B MESSAGE 11
                                ; 15     14     13     12     11     10     9      8
                                ;-----------------------------------------------------
                                ; PAR !  NU  !    NU      !   DECODED HEAD     ! SECTOR!
                                ;     !      !            !    ADDRESS         ! COUNT !
                                ;-----------------------------------------------------
                                ;
                                ;  7      6      5      4      3      2      1      0
                                ;-----------------------------------------------------
                                ;   SECTOR COUNT           ! UN ! UN !  1 !  1 !
                                ;-----------------------------------------------------
                                ;
                                .SBTTL  RK06 CONTROLLER REGISTER DEFINITION

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

```
000012          RKDS=   12              ;DRIVE STATUS REGISTER
000014          RKER=   14              ;ERROR REGISTER
000016          RKASOF= 16              ;ATTENTION SUMMARY AND OFFSET REGISTER
000020          RKDC=   20              ;DESIRED CYLINDER REGISTER
000020          RKDCYL= 20              ;DESIRED CYLINDER REGISTER
000024          RKDB=   24              ;DATA BUFFER
000026          RKMR1=  26              ;MAINTENANCE REGISTER 1
000034          RKMR2=  34              ;MAINTENANCE REGISTER 2
000036          RKMR3=  36              ;MAINTENANCE REGISTER 3
000030          RKPOS=  30              ;ECC POSITION INFORMATION
000030          RKECPS= 30              ;ECC POSITION INFORMATION
000032          RKPAT=  32              ;ECC PATTERN INFORMATION
000032          RKECPT= 32              ;ECC PATTERN INFORMATION

                .SBTTL  DRIVE COMMANDS

000101          SELDRV= 101             ;SELECT DRIVE
000103          PACK=   103             ;PACK ACKNOWLEDGE
000105          CLEAR=  105             ;DRIVE CLEAR
000107          UNLOAD= 107             ;UNLOAD
000111          SRTSPL= 111             ;START SPINDLE
000113          RECAL=  113             ;RECALIBRATE
000115          OFFSET= 115             ;OFFSET
000117          SEEK=   117             ;SEEK
000121          RDDATA= 121             ;READ DATA
000123          WRDATA= 123             ;WRITE DATA
000125          RDHEAD= 125             ;READ HEADER
000127          WRHEAD= 127             ;WRITE HEADER AND DATA
000131          WRTCHK= 131             ;WRITE CHECK

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

000140          RELEAS= 140             ;RELEASE DRIVE
000141          RDSTAT= 141             ;GET ALL STATUS FROM DRIVE
000164          RDALHD= 164             ;READ ALL HEADERS
000176          CONCLR= 176             ;CONTROLLER CLEAR (BIT 15 OF CS1)
000177          SUBCLR= 177             ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
000300          INTR=   300             ;GENERATE INTERRUPT TO CPU

                ;       DRIVER ISSUED SERVICE COMMANDS

000001          DR.SEL= 001             ;DRIVE SELECT
000005          DR.CLR= 005             ;DRIVE CLEAR

                .SBTTL  CONTROL AND STATUS REGISTER 1 BITS

000001          GO=     BIT0            ;GO BIT
000100          IE=     BIT6            ;INTERRUPT ENABLE
000200          RDY=    BIT7            ;CONTROLLER READY
000400          BA16=   BIT8            ;BUS ADDRESS BIT 16
001000          BA17=   BIT9            ;BUS ADDRESS BIT 17
002000          CDT=    BIT10           ;CONTROLLER DRIVE TYPE (0=RK06)
004000          CTO=    BIT11           ;CONTROLLER TIMED OUT WAITING FOR
                                        ;  DRIVE RESPONSE
010000          CFMT=   BIT12           ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
```

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

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

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

                                    .SBTTL  CONTROL AND STATUS REGISTER 2 BITS

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

                                    .SBTTL  ERROR REGISTER BIT DEFINITION

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

                                    .SBTTL  STATUS REGISTER BIT DEFINITION
```

```
                000001          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)
                004000          WRL=    BIT11                   ;WRITE LOCK
                020000          PIP=    BIT13                   ;POSITIONING IN PROGRESS
                040000          DSC=    BIT14                   ;DRIVE STATUS CHANGE
                100000          SVAL=   BIT15                   ;STATUS VALID

                                .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION

                000017          MESMSK= 17                      ;MESSAGE MASK

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

                                .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A

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

                                .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

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

```
                        DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B

3104    010000          S.DCLO= BIT12           ;DC LOW
3105    020000          S.DROT= BIT13           ;DRIVE OFF TRACK
3106    040000          S.UNS=  BIT14           ;DRIVE UNSAFE
3107
3108                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3109
3110    000020          S.XDOk= BIT4            ;TRANSDUCER OK
3111    000040          S.HDHM= BIT5            ;HEADS HOME
3112    000100          S.BRHM= BIT6            ;BRUSHES HOME
3113    000200          S.DOOR= BIT7            ;DOOR INTERLOCKED
3114    000400          S.CART= BIT8            ;CARTRAGE INTERLOCK
3115    001000          S.SPOK= BIT9            ;SPEED OK
3116    002000          S.FWD=  BIT10           ;FORWARD
3117    004000          S.REV=  BIT11           ;REVERSE
3118    010000          S.LOAD= BIT12           ;HEADS LOADING
3119    020000          S.RTZ=  BIT13           ;RETURN TO ZERO
3120    040000          S.UNLD= BIT14           ;HEADS UNLOADING
3121
3122                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3123
3124    000020          S.SECT= BIT4            ;SECTOR ERROR
3125    000040          S.WCLK= BIT5            ;WRITE CLOCK AND NO WRITE GATE
3126    000100          S.WGAT= BIT6            ;WRITE GATE AND NO TRANSISTIONS
3127    000200          S.HDFL= BIT7            ;HEAD FAULT
3128    000400          S.MHD=  BIT8            ;MULTIPLE HEAD SELECT
3129    001000          S.XERR= BIT9            ;INDEX ERROR
3130    002000          S.DIB=  BIT10           ;DIBIT ERROR
3131    004000          S.PLO=  BIT11           ;PLO ERROR
3132    010000          S.NMOV= BIT12           ;SEEK AND NO MOTION
3133    020000          S.LIMD= BIT13           ;LIMIT DETECT ON SEEK
3134    040000          S.BRKE= BIT14           ;SERVO-BRAKE
3135
3136                    .SBTTL  COMMON MASKS
3137
3138    000007          M.DRV=  7               ;DRIVE CODE
3139    100000          M.PAR=  BIT15           ;PARITY
3140    000003          M.ID=   3               ;BYTE ID
3141    017760          M.CDIF= 17760           ;CYLINDER DIFFERENCE/OFFSET
3142    017760          M.CADD= 17760           ;CYLINDER ADDRESS
3143    077770          M.SER=  77770           ;DRIVE SERIAL NUMBER
3144    000760          M.SECT= 760             ;SECTOR COUNT
3145    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 OFFSE"           ! 32
                        ;*   35 ! ERROR REGISTER                               ! 34
                        ;*   37 ! STATUS REGISTER                              ! 36
                        ;*   41 ! MESSAGE LINE A STATUS BYTE 00                ! 40
                        ;*   43 ! MESSAGE LINE B STATUS BYTE 00                ! 42
                        ;*   45 ! MESSAGE LINE A STATUS BYTE 01                ! 44
                        ;*   47 ! MESSAGE LINE B STATUS BYTE 01                ! 46
                        ;*   51 ! MESSAGE LINE A STATUS BYTE 10                ! 50
                        ;*   53 ! MESSAGE LINE B STATUS BYTE 10                ! 52
                        ;*   55 ! MESSAGE LINE A STATUS BYTE 11                ! 54
                        ;*   57 ! MESSAGE LINE B STATUS BYTE 11                ! 56
                        ;*   61 ! ECC POSITION INFORMATION                     ! 60
                        ;*   63 ! ECC PATTERN INFORMATION                      ! 62
                        ;*      ************************************************

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

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

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

                        .SBTTL  PROGRAM DEVICE STATUS REGISTER DEFINITION

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

```
3202            000100                  CMDTO=  BIT6                    ;NO TERMINATION TO COMMAND FOR AT
3203                                                                    ;  LEAST 1 SECOND
3204            000200                  W.WCK=  BIT7                    ;WRITE FOR WRITE WRITE CHECK
3205            000400                  NOCHK=  BIT8                    ;NO CHECK, DO NOT SET INTERRUPT ENABLE
3206            001000                  PBSVAL= BIT9                    ;PARAMETER STATUS WORDS VALID
3207                                                                    ;  (SET WHEN ERROR TERMINATION OR
3208                                                                    ;  READ STATUS COMMAND)
3209            002000                  DRPDRV= BIT10                   ;DROP DRIVE FROM TEST SEQUENCE
3210            004000                  NODSC=  BIT11                   ;ATTENTION SET BUT DCS AND FAULT RESET
3211            010000                  DRVSZD= BIT12                   ;DRIVE SEIZED BY OTHER PORT
3212            020000                  E.UNLD= BIT13                   ;DRIVE UNLOADED DUE TO ERROR
3213            040000                  Q.INIT= BIT14                   ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
3214            100000                  DTBAII= BIT15                   ;INHIBIT BUS ADDRESS INCREMENT
3215
3216                                    .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
3217
3218                                    ;       THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
3219                                    ;         FROM THE DRIVER TO THE CALLING PROGRAM
3220
3221            000016                  P.CS1=  16                      ;COMMAND AND STATUS REGISTER 1
3222            000020                  P.CS2=  20                      ;COMMAND AND STATUS REGISTER 2
3223            000022                  P.WCR=  22                      ;WORD COUNT REGISTER
3224            000024                  P.BAR=  24                      ;BUS ADDRESS REGISTER
3225            000026                  P.DTS=  26                      ;DESIRED TRACK SECTOR REGISTER
3226            000030                  P.DCYL= 30                      ;DESIRED CYLINDER REGISTER
3227            000032                  P.ASOF= 32                      ;ATTENTION SUMMARY/OFFSET REGISTER
3228            000034                  P.ER=   34                      ;ERROR REGISTER
3229            000036                  P.DS=   36                      ;STATUS REGISTER
3230            000040                  P.A00=  40                      ;MESSAGE A STATUS BYTE 00
3231            000042                  P.B00=  42                      ;MESSAGE B STATUS BYTE 00
3232            000044                  P.A01=  44                      ;MESSAGE A STATUS BYTE 01
3233            000046                  P.B01=  46                      ;MESSAGE B STATUS BYTE 01
3234            000050                  P.A10=  50                      ;MESSGGE A STATUS BYTE 10
3235            000052                  P.B10=  52                      ;MESSAGE B STATUS BYTE 10
3236            000054                  P.A11=  54                      ;MESSAGE A STATUS BYTE 11
3237            000056                  P.B11=  56                      ;MESSAGE B STATUS BYTE 11
3238            000060                  P.EPOS= 60                      ;ECC POSITION INFORMATION
3239            000062                  P.EPAT= 62                      ;ECC PATTERN INFORMATION
3240
3241                                    .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
3242
3243   002620     000                  PARM0:  .BYTE   0               ;DRIVE NUMBER
3244   002621     000                          .BYTE   0               ;COMMAND
3245   002622  000000                          .WORD   0               ;CYLINDER ADDRESS
3246   002624     000                          .BYTE   0               ;SECTOR ADDRESS
3247   002625     000                          .BYTE   0               ;TRACK ADDRESS
3248   002626     000                          .BYTE   0               ;OFFSET VALUE
3249   002627     000                          .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3250   002630  000000                          .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3251   002632  000000                          .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3252   002634  000000                          .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
3253   002636  000000                          .WORD   0               ;COMMAND AND STATUS REGISTER 1
3254   002640  000000                          .WORD   0               ;COMMAND AND STATUS REGISTER 2
3255   002642  000000                          .WORD   0               ;WORD COUNT REGISTER
3256   002644  000000                          .WORD   0               ;BUS ADDRESS REGISTER
3257   002646  000000                          .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
```

```
3258  002650  000000                    .WORD   0               ;DESIRED CYLINDER REGISTER
3259  002652  000000                    .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3260  002654  000000                    .WORD   0               ;ERROR REGISTER
3261  002656  000000                    .WORD   0               ;STATUS REGISTER
3262  002660  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3263  002662  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3264  002664  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3265  002666  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3266  002670  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3267  002672  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3268  002674  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3269  002676  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3270  002700  000000                    .WORD   0               ;ECC POSITION INFORMATION
3271  002702  000000                    .WORD   0               ;ECC PATTERN INFORMATION
3272
3273                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
3274
3275  002704     000         PARM1:  .BYTE   0               ;DRIVE NUMBER
3276  002705     000                 .BYTE   0               ;COMMAND
3277  002706  000000                 .WORD   0               ;CYLINDER ADDRESS
3278  002710     000                 .BYTE   0               ;SECTOR ADDRESS
3279  002711     000                 .BYTE   0               ;TRACK ADDRESS
3280  002712     000                 .BYTE   0               ;OFFSET VALUE
3281  002713     000                 .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3282  002714  000000                 .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3283  002716  000000                 .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3284  002720  000000                 .WORD   0               ;PROGRAM DRIVE STATJS INFORMATION
3285  002722  000000                 .WORD   0               ;COMMAND AND STATUS REGISTER 1
3286  002724  000000                 .WORD   0               ;COMMAND AND STATUS REGISTER 2
3287  002726  000000                 .WORD   0               ;WORD COUNT REGISTER
3288  002730  000000                 .WORD   0               ;BUS ADDRESS REGISTER
3289  002732  000000                 .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
3290  002734  000000                 .WORD   0               ;DESIRED CYLINDER REGISTER
3291  002736  000000                 .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3292  002740  000000                 .WORD   0               ;ERROR REGISTER
3293  002742  000000                 .WORD   0               ;STATUS REGISTER
3294  002744  000000                 .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3295  002746  000000                 .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3296  002750  000000                 .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3297  002752  000000                 .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3298  002754  000000                 .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3299  002756  000000                 .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3300  002760  000000                 .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3301  002762  000000                 .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3302  002764  000000                 .WORD   0               ;ECC POSITION INFORMATION
3303  002766  000000                 .WORD   0               ;ECC PATTERN INFORMATION
3304
3305                          .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
3306
3307  002770  000000         T.CS1:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3308                                                         ;  REGISTER 1
3309  002772  000000         T.CS2:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3310                                                         ;  REGISTER 2
3311  002774  000000         T.WCR:  .WORD   0               ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
3312  002776  000000         T.BA:   .WORD   0               ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
3313  003000  000000         T.DA:   .WORD   0               ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
003002  000000  T.DC:   .WORD   0       ;TEMPORARY STORAGE FOR DRIVE CYLINDER
003004  000000  T.ASOF: .WORD   0       ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
                                        ;  AND OFFSET
003006  000000  T.ER:   .WORD   0       ;TEMPORARY STORAGE FOR ERROR REGISTER
003010  000000  T.DS:   .WORD   0       ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
003012  000000  T.MR1:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
003014  000000  T.MR2:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
003016  000000  T.MR3:  .WORD   0       ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
003020  000000  T.POS:  .WORD   0       ;TEMPORARY STORAGE FOR ECC POSITION
003022  000000  T.PAT:  .WORD   0       ;TEMPORARY STORAGE FOR ECC PATTERN
003024  000000  T.DB:   .WORD   0       ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER

                .SBTTL  DRIVER PARAMETERS

003026  177440  RKBAS:  .WORD   177440  ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
003030  000210  RKVEC:  .WORD   210     ;ADDRESS OF R611 VECTOR
003032  000240  RKPRI:  .WORD   PR5     ;RK611 INTERRUPT PRIORITY
003034  240106  A.NORM: ERRFRE          ;ADDRESS OF NORMAL RETURN FROM DRIVER
003036  241354  A.ABNL: ERRHDL          ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
003040  240650  A.CONT: CONERR          ;ADDRESS OF CONTROLLER ERROR RETURN
003042  000000  E.CONT: .WORD   0       ;CONTROLLER ERROR STATUS
                                        ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
                                        ;  IS INITIATED.  IF A CONTROLLER ERROR
                                        ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
                                        ;  USED:
                                        ;
000000  E.CCLR= BIT0                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
000001  E.NOAT= BIT1                    ;NO ATTENTION IN ATTENTION SUMMARY REG
000002  E.UATT= BIT2                    ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
000004  E.UDAT= BIT3                    ;UNEXPECTED DATA TYPE ERROR
000010  E.CLAT= BIT4                    ;ATTENTION DID NOT RESET WITH CLEAR
000020  E.SCLR= BIT5                    ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
                                        ;  ATTENTION
000100  E.ILLD= BIT6                    ;ILLEGAL DRIVER COMMAND
000400  E.LTY=  BIT8                    ;DATA LATE WHEN UNLOADING HEADER
001000  E.CERR= BIT9                    ;CONTROLLER ERROR DURING DRIVER SERVICING
002000  E.DPAR= BIT10                   ;DRIVE DETECTED PARITY ERROR
004000  E.CMTO= BIT11                   ;CONTROLLER COMMAND TIME OUT (QUELED ONLY)
100000  E.MDS=  BIT15                   ;MULTIPLE DRIVE SELECT

003044  000000  C.WAIT: .WORD   0       ;PARAMETER BLOCK OF THE DRIVE
                                        ;WAITING FOR COMMAND COMPLETION
003046  000400  W.MTIM: .WORD   400     ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
003050  000000  W.MILI: .WORD   400     ;16 MILISECOND TIME FOR PROGRAM
                                        ;
                                        ;       CPU     VALUE
                                        ;       ---     -----
                                        ;
                                        ;       11/05   100
                                        ;       11/10
                                        ;       11/20
                                        ;       11/34
                                        ;       11/40
                                        ;       11/45   400
                                        ;       11/50
                                        ;       11/70
```

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

                        .SBTTL  INTERRUPT MASKS

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

                        ;       INTERRUPT MASK TABLE

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

                        .SBTTL  PARAMETER BLOCK TABLE

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

                        .SBTTL  TIME FOR WATCH-DOG TIMER

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

PROGRAM SPECIFIC RESERVED LOCATIONS

```
003136  000000          DLTFLG: .BYTE   0        ;DATA LATE ERROR FLAG
003137  000000          NORTRY: .BYTE   0        ;"NO-RETRY" FLAG
003134  000000          JBMPRS: .BYTE   0        ;UNIBUS MAP PRESENT IF = 1
003135  000000          MEMABT: .BYTE   0        ;SET BYTE = 1 FOR NO ABORT
                                                 ;    ON MEMORY PARITY ERRORS
003136  000000          HLPOVL: .BYTE   0        ;HELP FILE OVERLAID INDICATOR
003137  000000          NOTYPE: .BYTE   0        ;INDICATES PROGRAM JUST LOADED IF 0
                        WHDSW=BIT0               ;WRITE HEADER & DATA SWITCH
                        VFHDSW=BIT1              ;VERIFY HEADERS SWITCH
                        WCDASW=BIT2              ;WRITE CHECK DATA SWITCH
                        RCDASW=BIT3              ;READ CHECK DATA SWITCH
                        OREQSW=BIT4              ;OFFSET REQUIRED SWITCH
                                .EVEN

003140  177546          LKS:    .WORD   177546   ;KW11-L CLOCK STATUS REGISTER
003142  172540          PKS:    .WORD   172540   ;KW11-P CONTROL AND STATUS REGISTER
003144  172542          PKSB:   .WORD   172542   ;KW11-P COUNT SET BUFFER REGISTER
003146  172544          PKRB:   .WORD   172544   ;KW11-P COUNTER REGISTER
003150  000100          LCVEC:  .WORD   100      ;KW11-L VECTOR STORAGE
003152  000104          PCVEC:  .WORD   104      ;KW11-P VECTOR STORAGE
                        MEMVEC=114               ;MEMORY PARITY TRAP VECTOR
003154  000000          TCONLO: .WORD   0        ;LO BITS OF CALIBRATION TIME CONSTANT
003156  000000          TCONHI: .WORD   0        ;HI BITS OF CALIB TIME CONST
003160  000000          BLWMIN: .WORD   0        ;COUNT OF TIMES BELOW MIN
003162  000000          ABVMX1: .WORD   0        ;COUNT OF FORWARD TIMES ABOVE MAX
003164  000000          ABVMX2: .WORD   0        ;COUNT OF REVERSE TIMES ABOVE MAX
003166  000000          SUMLO1: .WORD   0        ;LO BITS OF FORWARD TIME SUM
003170  000000          SUMHI1: .WORD   0        ;HI BITS OF FORWARD TIME SUM
003172  000000          SUMLO2: .WORD   0        ;LO BITS OF REVERSE TIME SUM
003174  000000          SUMHI2: .WORD   0        ;HI BITS OF REVERSE TIME SUM
003176  000000          SAVPAR: .WORD   0        ;SAVE WORD FOR PAR CONSTANT
003200  000000          SAVWRD: .WORD   0        ;SCRATCH WORD
003202  000000          MINIL1: .WORD   0        ;LO BITS OF MIN MEASURED TIME (FORWARD)
003204  000000          MINIH1: .WORD   0        ;HI BITS OF MIN MEASURED TIME (FORWARD)
003206  000000          MAXIL1: .WORD   0        ;LO BITS OF MAX MEASURED TIME (FORWARD)
003210  000000          MAXIH1: .WORD   0        ;HI BITS OF MAX MEASURED TIME (FORWARD)
003212  000000          MINIL2: .WORD   0        ;LO BITS OF MIN MEASURED TIME (REVERSE)
003214  000000          MINIH2: .WORD   0        ;HI BITS OF MIN MEASURED TIME (REVERSE)
003216  000000          MAXIL2: .WORD   0        ;LO BITS OF MAX MEASURED TIME (REVERSE)
003220  000000          MAXIH2: .WORD   0        ;HI BITS OF MAX MEASURED TIME (REVERSE)
003222  000400          BSSOFT: .BLKW   ^D256    ;RECORD OF BAD SECTORS FROM SOFTWARE
004222  000400          BSFACT: .BLKW   ^D256    ;RECORD OF BAD SECTORS FROM FACTORY
005222  000006          PRVCMD: .BLKW   6        ;PREVIOUS COMMAND STORAGE
005236  000006          COMSTR: .BLKW   6        ;CURRENT COMMAND STORAGE
005252  000000          PRMPLO: .WORD   0        ;PREV. U.B. MAP REG 0
005254  000000          PRMPHO: .WORD   0
005256  000000          CRMPLO: .WORD   0        ;CURRENT U.B. MAP REG 0
005260  000000          CRMPHO: .WORD   0
005262  000102          BUFFO:  .BLKW   ^D66     ;OUTPUT BUFFER 1
005466  000000          LOWOCT: .WORD   0        ;LOW 16 BITS OF CONVERTED BINARY NUMBER
005470  000000          HIGOCT: .WORD   0        ;HIGH BITS OF CONVERTED BINARY NO.
005472  000000          BUFPRT: .WORD   0        ;BUFFER POINTER
005474  000000          RECODE: .WORD   0        ;RECOVERY CODE WORD
005476  000000          ERRCOM: .WORD   0        ;ERROR COMMAND
005500  000000          DRIVE:  .WORD   0        ;NO.  OF  DRIVE IN USE
005502  000000          STALLS: .WORD   0        ;CURRENT NO. OF UNIT STALLS TO APPLY
```

```
                        PROGRAM SPECIFIC RESERVED LOCATIONS

005504  000000  CYLNDR: .WORD   0               ;CURRENT CYLINDER NUMBER
005506  000000  FS:     .WORD   0               ;FIRST SECTOR LIMIT
005510  000000  LS:     .WORD   0               ;LAST SECTOR LIMIT
005512  000000  NCYL1:  .WORD   0               ;NEXT CYL SCRATCH WORD
005514  000000  NCYL2:  .WORD   0               ;NEXT CYL SCRATCH WORD
005516  000000  OFINUS: .WORD   0               ;OFFSET IN USE
005520  000000  TRACK:  .WORD   0               ;TRACK IN USE
005522  000000  INTCHR: .WORD   0               ;TTY INTERRUPT INPUT WORD
005524  000000  SELECT: .WORD   0               ;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
005526  000000  ONLINE: .WORD   0               ;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
005530  000000  NEWON:  .WORD   0               ;NEW LOOK AT ONLINE DRIVES
005532  000000  SCRACH: .WORD   0               ;GENERAL PURPOSE SCRATCH WORD
005534  000000  PATRN:  .WORD   0               ;USER PATTERN LIST WORD
005536  000000  XXDPAD: .WORD   0               ;STARTING ADDRESS OF XXDP LOADER
005540  000002  XDPSAV: .BLKW   2               ;XXDP PHYSICAL SAVE ADDRESS
005544  000000  PLOFST: .WORD   0               ;+ OFFSET VALUE
005546  000000  NGOFST: .WORD   0               ;- OFFSET VALUE
005550  000005  SAVPRS: .BLKW   5               ;SAVE INITIAL PARAMS FOR XFER
005562  000000  WDSXFR: .WORD   0               ;NO. OF WORDS ACTUALLY XFERRED
005564  000000  FINCYL: .WORD   0               ;FINAL CYLINDER ADRS
005566     000  FINTRK: .BYTE   0               ;FINAL TRACK ADRS
005567     000  FINSEC: .BYTE   0               ;FINAL SECTOR ADRS
005570  000000  LASTWC: .WORD   0               ;ACTUAL FINAL WC
005572  000002  PMA:    .BLKW   2               ;PARTIAL TRANSFER MEMORY START ADRS


                        ;LIST OF DESIRED DRIVES TO TEST (IF BYTE=0, DON'T TEST DRIVE:
005576                  DRVLST:
005576     001          .BYTE   1               ;DRIVE 0
005577     001          .BYTE   1               ;DRIVE 1
005600     001          .BYTE   1               ;DRIVE 2
005601     001          .BYTE   1               ;DRIVE 3
005602     001          .BYTE   1               ;DRIVE 4
005603     001          .BYTE   1               ;DRIVE 5
005604     001          .BYTE   1               ;DRIVE 6
005605     001          .BYTE   1               ;DRIVE 7


                        ;LIST OF ITERATION COUNTS FOR DESIRED TESTS (IF=0, TEST NOT RUN)
                        ;MAXIMUM ALLOWABLE NUMBER OF ITERATIONS = 77776 (OCT)
005606                  TSTLST:
005606  000010          .WORD   10              ;TEST 01
005610  000200          .WORD   200             ;TEST 02
005612  000010          .WORD   10              ;TEST 03
005614  000010          .WORD   10              ;TEST 04
005616  000002          .WORD   2               ;TEST 05
005620  000002          .WORD   2               ;TEST 06
005622  000002          .WORD   2               ;TEST 07
005624  000400          .WORD   400             ;TEST 10
005626  000500          .WORD   500             ;TEST 11
005630  000002          .WORD   2               ;TEST 12
005632  000001          .WORD   1               ;TEST 13
005634  000001          .WORD   1               ;TEST 14
005636  000001          .WORD   1               ;TEST 15
```

```
                                PROGRAM SPECIFIC RESERVED LOCATIONS

003638  005640  000001                .WORD   1               ;TEST 16
003640  005642  000010                .WORD   10              ;TEST 17
003642  005644  000010                .WORD   10              ;TEST 20
003644  005646  000001                .WORD   1               ;TEST 21


                                ;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
                                DFLTST:
003650  005650                        .WORD   10              ;TEST 01
003652  005652  000010                .WORD   200             ;TEST 02
003654  005654  000200                .WORD   10              ;TEST 03
003656  005656  000010                .WORD   10              ;TEST 04
003660  005660  000010                .WORD   2               ;TEST 05
003662  005662  000002                .WORD   2               ;TEST 06
003664  005664  000002                .WORD   2               ;TEST 07
003666  005666  000002                .WORD   400             ;TEST 10
003670  005670  000400                .WORD   500             ;TEST 11
003672  005672  000500                .WORD   2               ;TEST 12
003674  005674  000002                .WORD   1               ;TEST 13
003676  005676  000001                .WORD   1               ;TEST 14
003700  005700  000001                .WORD   1               ;TEST 15
003702  005702  000001                .WORD   1               ;TEST 16
003704  005704  000000                .WORD   10              ;TEST 17
003706  005706  000010                .WORD   10              ;TEST 20
003710  005710  000010                .WORD   1               ;TEST 21


                000021          NMTSTS=(DFLTST-TSTLST)/2        ;TOTAL NO. OF AUTOMATIC TESTS


                                ;OPERATING PARAMETER LIST
                                PRMLST:
003712  005712                  FC:     .WORD   0               ;FIRST CYLINDER
003714  005712  000000          LC:     .WORD   632             ;LAST CYLINDER
003716  005714  000632          IC:     .WORD   1               ;CYLINDER INCREMENT
003720  005716  000001          FT:     .WORD   0               ;FIRST TRACK
003722  005720  000000          LT:     .WORD   2               ;LAST TRACK
003724  005722  000002          IT:     .WORD   1               ;TRACK INCREMENT
003726  005724  000001          SO:     .WORD   0               ;FIRST SECTOR IF 20(DEC) SECTOR FMT
003730  005726  000000          S1:     .WORD   23              ;LAST SECTOR IF 20(DEC) SECTOR FMT
003732  005730  000023          S2:     .WORD   0               ;FIRST SECTOR IF 22(DEC) SECTOR FMT
003734  005732  000000          S3:     .WORD   25              ;LAST SECTOR IF 22(DEC) SECTOR FMT
003736  005734  000025          IS:     .WORD   1               ;SECTOR INCREMENT
003740  005736  000001          PT:     .WORD   0               ;DATA PATTERN SELECT WORD
003742  005740  000000          MA:     .WORD   RWBUF           ;LO BITS OF PHYS. MEM. ADDR. (BITS 0-15)
003744  005742  063526                  .WORD   0               ;HI BITS OF PHYS. MEM. ADDR. (BITS 16-21)
003746  005744  000000          WC:     .WORD   403             ;WORD COUNT (IF WC=0, WRD CNT IS 65,536 DEC)
003750  005746  000403          CS:     .WORD   0               ;CONTROL SWITCH WORD
003752  005750  000000          ST:     .WORD   0               ;NUMBER OF UNIT STALLS
003754  005752  000000          SM:     .WORD   1000            ;MAX. STALLS, TEST 11
003756  005754  001000


                                ;DEFAULT OPERATING PARAMETER VALUES
                                PRDFLT:
003760  005756                          .WORD   0               ;FC DEFAULT
```

PROGRAM SPECIFIC RESERVED LOCATIONS

```
                        .WORD   632             ;LC DEFAULT
                        .WORD   1               ;IC DEFAULT
                        .WORD   0               ;FT DEFAULT
                        .WORD   2               ;LT DEFAULT
                        .WORD   1               ;IT DEFAULT
                        .WORD   0               ;S0 DEFAULT
                        .WORD   23              ;S1 DEFAULT
                        .WORD   0               ;S2 DEFAULT
                        .WORD   25              ;S3 DEFAULT
                        .WORD   1               ;IS DEFAULT
                        .WORD   0               ;PT DEFAULT
                        .WORD   RWBUF           ;LOW MA (0-15) DEFAULT
                        .WORD   0               ;HIGH MA (16-21) DEFAULT
                        .WORD   403             ;WC DEFAULT
                        .WORD   0               ;CS DEFAULT
                        .WORD   0               ;ST DEFAULT
                        .WORD   1000            ;SM DEFAULT


;OPERATING PARAMETER VALUE LOW AND HIGH LIMITS
PRMLIM:
                        .WORD   0               ;FC LIMITS
                        .WORD   631
                        .WORD   0               ;LC LIMITS
                        .WORD   632
                        .WORD   1               ;IC LIMITS
                        .WORD   632
                        .WORD   0               ;FT LIMITS
                        .WORD   2
                        .WORD   0               ;LT LIMITS
                        .WORD   2
                        .WORD   1               ;IT LIMITS
                        .WORD   2
                        .WORD   0               ;S0 LIMITS
                        .WORD   23
                        .WORD   0               ;S1 LIMITS
                        .WORD   23
                        .WORD   0               ;S2 LIMITS
                        .WORD   25
                        .WORD   0               ;S3 LIMITS
                        .WORD   25
                        .WORD   1               ;IS LIMITS
                        .WORD   25
                        .WORD   0               ;PT LIMITS
                        .WORD   177777
                        .WORD   RWBUF           ;MA LOWER LIMIT
                        .WORD   0
MAHILM:                 .WORD   157776          ;MA UPPER LIMIT
                        .WORD   77
                        .WORD   0               ;WC LIMITS
                        .WORD   177777
                        .WORD   0               ;CS LIMITS
                        .WORD   000070
                        .WORD   0               ;ST LIMITS
                        .WORD   177777
```

```
                        PROGRAM SPECIFIC RESERVED LOCATIONS

3650  006126  000000            .WORD   0               ;SM LIMITS
3651  006130  177777            .WORD   177777
3652
3653
3654
3655
3656  006132                    ;ASCII PARAMETER MNEMONICS
3657  006132  041503            PRMNEM:
3658  006134  041514                    .ASCII  /FC/
3659  006136  042111                    .ASCII  /LC/
3660  006140  052124                    .ASCII  /IC/
3661  006142  052114                    .ASCII  /FT/
3662  006144  052111                    .ASCII  /LT/
3663  006146  030123                    .ASCII  /IT/
3664  006150  030523                    .ASCII  /S0/
3665  006152  031123                    .ASCII  /S1/
3666  006154  031523                    .ASCII  /S2/
3667  006156  051511                    .ASCII  /S3/
3668  006160  052120                    .ASCII  /IS/
3669  006162  040515                    .ASCII  /PT/
3670  006164  000000                    .ASCII  /MA/
3671  006166  041527                    .WORD   0               ;TABLE FILLER FOR MA
3672  006170  051503                    .ASCII  /WC/
3673  006172  052123                    .ASCII  /CS/
3674  006174  046523                    .ASCII  /ST/
3675                                    .ASCII  /SM/
3676
3677
3678                        ;DATA PATTERN CC
3679                        ;       HI-LO FREQ. MIX
3680  006176                PATCC:
3681  006176  177777                   177777
3682  006200  177777                   177777
3683  006202  177777                   177777
3684  006204  052525                   052525
3685  006206  052525                   052525
3686  006210  052525                   052525
3687  006212  177777                   177777
3688  006214  177777                   177777
3689  006216  052525                   052525
3690  006220  052525                   052525
3691  006222  177777                   177777
3692  006224  052525                   052525
3693  006226  177252                   177252
3694  006230  177252                   177252
3695  006232  172765                   172765
3696  006234  172765                   172765
3697
3698
3699
3700
3701                        ;DATA PATTERN 01
3702  006236                ;       HI FREQ. PHASE MIX
3703  006236  000000            PAT01:
3704  006240  000000                   000000
3705  006242  000000                   000000
                                       000000
```

```
3706  006244  177777                  177777
3707  006246  177777                  177777
3708  006250  177777                  177777
3709  006252  000000                  000000
3710  006254  000000                  000000
3711  006256  177777                  177777
3712  006260  177777                  177777
3713  006262  000000                  000000
3714  006264  177777                  177777
3715  006266  000000                  000000
3716  006270  177777                  177777
3717  006272  000000                  000000
3718  006274  177777                  177777
3719
3720
3721
3722                          ;DATA PATTERN 02
3723                          ;       LO FREQ. PHASE MIX
3724  006276                  PAT02:
3725  006276  052525                  052525
3726  006300  052525                  052525
3727  006302  052525                  052525
3728  006304  125252                  125252
3729  006306  125252                  125252
3730  006310  125252                  125252
3731  006312  052525                  052525
3732  006314  052525                  052525
3733  006316  125252                  125252
3734  006320  125252                  125252
3735  006322  052525                  052525
3736  006324  125252                  125252
3737  006326  052525                  052525
3738  006330  125252                  125252
3739  006332  052525                  052525
3740  006334  125252                  125252
3741
3742
3743
3744                          ;DATA PATTERN 03
3745                          ;       MAX. PRECOMP. PHASE MIX
3746  006336                  PAT03:
3747  006336  133333                  133333
3748  006340  066666                  066666
3749  006342  155555                  155555
3750  006344  155555                  155555
3751  006346  133333                  133333
3752  006350  066666                  066666
3753  006352  066666                  066666
3754  006354  155555                  155555
3755  006356  155555                  155555
3756  006360  133333                  133333
3757  006362  133333                  133333
3758  006364  133333                  133333
3759  006366  133333                  133333
3760  006370  133333                  133333
3761  006372  133333                  133333
```

```
3762  006374  133333                          133333
3763
3764
3765
3766                                  ;DATA PATTERN 04
3767                                  ;       ROTATING BOUNDARY PULSE PRECOMP.
3768  006376                          PAT04:
3769  006376  121105                          121105
3770  006400  150442                          150442
3771  006402  064221                          064221
3772  006404  132110                          132110
3773  006406  055044                          055044
3774  006410  026422                          026422
3775  006412  013211                          013211
3776  006414  105504                          105504
3777  006416  042642                          042642
3778  006420  021321                          021321
3779  006422  110550                          110550
3780  006424  044264                          044264
3781  006426  022132                          022132
3782  006430  011055                          011055
3783  006432  104426                          104426
3784  006434  042213                          042213
3785
3786
3787
3788                                  ;DATA PATTERN 05
3789                                  ;       ROTATING CELL PULSE PRECOMP.
3790  006436                          PAT05:
3791  006436  026455                          026455
3792  006440  113226                          113226
3793  006442  045513                          045513
3794  006444  122645                          122645
3795  006446  151322                          151322
3796  006450  064551                          064551
3797  006452  132264                          132264
3798  006454  055132                          055132
3799  006456  026455                          026455
3800  006460  113226                          113226
3801  006462  045513                          045513
3802  006464  122645                          122645
3803  006466  151322                          151322
3804  006470  064551                          064551
3805  006472  132264                          132264
3806  006474  055132                          055132
3807
3808
3809                                  ;DATA PATTERN 06
3810                                  ;       ALL ZEROS
3811  006476                          PAT06:
3812  006476  000000                          000000
3813  006500  000000                          000000
3814  006502  000000                          000000
3815  006504  000000                          000000
3816  006506  000000                          000000
3817  006510  000000                          000000
```

```
3818  006512  000000                  000000
3819  006514  000000                  000000
3820  006516  000000                  000000
3821  006520  000000                  000000
3822  006522  000000                  000000
3823  006524  000000                  000000
3824  006526  000000                  000000
3825  006530  000000                  000000
3826  006532  000000                  000000
3827  006534  000000                  000000
3828
3829
3830
3831                          ;DATA PATTERN 07
3832                          ;       ALL ONES
3833  006536                  PAT07:
3834  006536  177777                  177777
3835  006540  177777                  177777
3836  006542  177777                  177777
3837  006544  177777                  177777
3838  006546  177777                  177777
3839  006550  177777                  177777
3840  006552  177777                  177777
3841  006554  177777                  177777
3842  006556  177777                  177777
3843  006560  177777                  177777
3844  006562  177777                  177777
3845  006564  177777                  177777
3846  006566  177777                  177777
3847  006570  177777                  177777
3848  006572  177777                  177777
3849  006574  177777                  177777
3850
3851
3852
3853                          ;DATA PATTERN 08
3854  006576                  ;       SHIFTED 1 IN FIELD OF ZEROS
3855  006576  000001          PAT08:
3856  006600  000002                  000001
3857  006602  000004                  000002
3858  006604  000010                  000004
3859  006606  000020                  000010
3860  006610  000040                  000020
3861  006612  000100                  000040
3862  006614  000200                  000100
3863  006616  000400                  000200
3864  006620  001000                  000400
3865  006622  002000                  001000
3866  006624  004000                  002000
3867  006626  010000                  004000
3868  006630  020000                  010000
3869  006632  040000                  020000
3870  006634  100000                  040000
3871                                  100000
3872
3873
```

```
3874
3875
3876  006636
3877  006636  177776          ;DATA PATTERN 09
3878  006640  177775          ;       SHIFTED 0 IN FIELD OF ONES
3879  006642  177773          PAT09:
3880  006644  177767                  177776
3881  006646  177757                  177775
3882  006650  177737                  177773
3883  006652  177677                  177767
3884  006654  177577                  177757
3885  006656  177377                  177737
3886  006660  176777                  177677
3887  006662  175777                  177577
3888  006664  173777                  177377
3889  006666  167777                  176777
3890  006670  157777                  175777
3891  006672  137777                  173777
3892  006674  077777                  167777
3893                                  157777
3894                                  137777
3895                                  077777
3896
3897                          ;DATA PATTERN 10
3898  006676                  ;       ALTERNATING 0-1
3899  006676  052525          PAT10:
3900  006700  052525                  052525
3901  006702  052525                  052525
3902  006704  052525                  052525
3903  006706  052525                  052525
3904  006710  052525                  052525
3905  006712  052525                  052525
3906  006714  052525                  052525
3907  006716  052525                  052525
3908  006720  052525                  052525
3909  006722  052525                  052525
3910  006724  052525                  052525
3911  006726  052525                  052525
3912  006730  052525                  052525
3913  006732  052525                  052525
3914  006734  052525                  052525
3915
3916
3917
3918                          ;DATA PATTERN 11
3919                          ;       ALTERNATING 1-0
3920  006736                  PAT11:
3921  006736  125252                  125252
3922  006740  125252                  125252
3923  006742  125252                  125252
3924  006744  125252                  125252
3925  006746  125252                  125252
3926  006750  125252                  125252
3927  006752  125252                  125252
3928  006754  125252                  125252
3929  006756  125252                  125252
```

```
3930  006760  125252                  125252
3931  006762  125252                  125252
3932  006764  125252                  125252
3933  006766  125252                  125252
3934  006770  125252                  125252
3935  006772  125252                  125252
3936  006774  125252                  125252
3937
3938
3939
3940                                  ;DATA PATTERN 12
3941                                  ;       SHIFTING ZEROS AND ONES
3942  006776                          PAT12:
3943  006776  000001                  000001
3944  007000  000003                  000003
3945  007002  000007                  000007
3946  007004  000017                  000017
3947  007006  000037                  000037
3948  007010  000077                  000077
3949  007012  000177                  000177
3950  007014  000377                  000377
3951  007016  000777                  000777
3952  007020  001777                  001777
3953  007022  003777                  003777
3954  007024  007777                  007777
3955  007026  017777                  017777
3956  007030  037777                  037777
3957  007032  077777                  077777
3958  007034  177777                  177777
3959
3960
3961
3962                                  ;DATA PATTERN 13
3963                                  ;       COMPOSITE ROTATING
3964  007036                          PAT13:
3965  007036  072307                  072307
3966  007040  135143                  135143
3967  007042  156461                  156461
3968  007044  167230                  167230
3969  007046  073514                  073514
3970  007050  035646                  035646
3971  007052  016723                  016723
3972  007054  107351                  107351
3973  007056  143564                  143564
3974  007060  061672                  061672
3975  007062  030735                  030735
3976  007064  114356                  114356
3977  007066  046167                  046167
3978  007070  123073                  123073
3979  007072  151453                  151453
3980  007074  164616                  164616
3981
3982
3983
3984                                  ;DATA PATTERN 14
3985                                  ;       PSEUDO-RANDOM (COMPUTED BY PROGRAM)
```

```
3986  007076                          PAT14:
3987  007076  000000                            .WORD
3988  007100  000000                            .WORD
3989  007102  000000                            .WORD
3990  007104  000000                            .WORD
3991  007106  000000                            .WORD
3992  007110  000000                            .WORD
3993  007112  000000                            .WORD
3994  007114  000000                            .WORD
3995  007116  000000                            .WORD
3996  007120  000000                            .WORD
3997  007122  000000                            .WORD
3998  007124  000000                            .WORD
3999  007126  000000                            .WORD
4000  007130  000000                            .WORD
4001  007132  000000                            .WORD
4002  007134  000000                            .WORD
4003
4004
4005                                  ;USER-DEFINED DATA PATTERN 15
4006  007136                          PAT15:
4007  007136  072307                            072307                  ;WORD 00
4008  007140  135143                            135143                  ;WORD 01
4009  007142  156461                            156461                  ;WORD 02
4010  007144  167230                            167230                  ;WORD 03
4011  007146  073514                            073514                  ;WORD 04
4012  007150  035646                            035646                  ;WORD 05
4013  007152  016723                            016723                  ;WORD 06
4014  007154  107351                            107351                  ;WORD 07
4015  007156  143564                            143564                  ;WORD 10
4016  007160  061672                            061672                  ;WORD 11
4017  007162  030735                            030735                  ;WORD 12
4018  007164  114356                            114356                  ;WORD 13
4019  007166  046167                            046167                  ;WORD 14
4020  007170  123073                            123073                  ;WORD 15
4021  007172  151453                            151453                  ;WORD 16
4022  007174  164616                            164616                  ;WORD 17
4023
4024
4025
4026          057467          MINL01=^D24375    ;LO BITS OF SPEC'D MIN ROT. LATENCY
4027          000000          MINHI1=0          ;HI BITS OF SPEC'D MIN ROT. LATENCY
4028          062031          MAXL01=^D25625    ;LO BITS OF SPEC'D MAX ROT. LATENCY
4029          000000          MAXHI1=0          ;HI BITS OF SPEC'D MAX ROT. LATENCY
4030          017500          MAXL02=^D8000     ;LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
4031          000000          MAXHI2=0          ;HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
4032          112160          MAXL03=^D38000    ;LO BITS OF SPEC'D MAX AVG SEEK TIME
4033          000000          MAXHI3=0          ;HI BITS OF SPEC'D MAX AVG SEEK TIME
4034          022370          MAXL04=^D9464     ;LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
4035          000001          MAXHI4=1          ;HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
4036          002734          RNDSHT=^D1500     ;SHORT RANDOM SEEKS = 1500(10)
4037          016514          RNDLNG=^D7500     ;LONG RANDOM SEEKS = 7500(10)
4038          000632          LSTCYL=632        ;LAST CYL. = 410(10)
4039          000002          LSTTRK=2          ;LAST TRACK  = 2
4040          000365          ALNCYL=365        ;ALIGNMENT CYLINDER =245(10)
4041          000002          BSERR=BIT1        ;BSE ERROR
```

```
                    000004          HVRCER=BIT2                     ;HVRC ERROR
                    000010          OPIERR=BIT3                     ;OPI ERROR
                    000020          DCKERR=BIT4                     ;DATA CHECK ERROR
                    000040          ECCNC=BIT5                      ;ECC NON-CORRECTABLE
                    000100          WCERR=BIT6                      ;WRITE CHECK ERROR
                    000200          ABORT=BIT7                      ;ABORT
                    000400          LEV2ER=BIT8                     ;LEVEL TWO ERROR
                    001000          BADSEC=BIT9                     ;BAD SECTOR FLAG
                    002000          TWOTOS=BIT10                    ;TWO TIME OUTS
                    004000          RCL_REQ=BIT11                   ;RECALIBRATE REQUIRED
                    010000          DRNAVL=BIT12                    ;DRIVE NOT RELSD BY OTHER PORT
                    100000          ANYDER=BIT15                    ;ANY ERROR DETECTED FLAG

4055  007176  005015  042115  033455  DZR6M:  .ASCIZ  <15><12>/MD-11-DZR6M-C/
4056  007204  026461  055104  033122
4057  007212  026515  000103
4058  007216  026440  051040  033113  SUBVER: .ASCIZ  & - RK611/RK06 SUBSYSTEM VERIFICATION : PART 3
4059  007224  030461  051057  030113
4060  007232  020066  052523  051502
4061  007240  020531  042524  020115
4062  007246  042526  044522  044506
4063  007254  040503  044524  047117
4064  007262  035040  050040  051101
4065  007270  020124  000000
4066  007273  000061  005015  000000  PART1:  .ASCIZ  /1/<15><12>
4067  007277  000062  005015  000000  PART2:  .ASCIZ  /2/<15><12>
4068  007303  000015  005012  040514  LSTMEM: .ASCIZ  <15><12><12>/LAST PHYS MEM ADR = /
4069  007310  052123  050040  054510
4070  007316  020123  042515  020115
4071  007324  042101  020122  020075
4072  007332  000000
4073  007333  000117  042526  046122  OVLODR: .ASCIZ  /OVERLAY LOADER ? (Y OR N) * /
4074  007340  054501  046040  040517
4075  007346  042504  020122  020077
4076  007354  054450  047440  020122
4077  007362  024516  025040  000040
4078  007370  005015  040520  040522  PRMINP: .ASCIZ  <15><12>/PARAMETER INPUT MODE/<15><12><12>
4079  007376  042515  042524  020122
4080  007404  047111  052520  020124
4081  007412  047515  042504  005015
4082  007420  000012
4083  007422  045522  033060  041040  RkBADR: .ASCIZ  /RK06 BUS ADR = /
4084  007430  051525  040440  051104
4085  007436  036440  000040
4086  007442  045522  033060  053040  RKVADR: .ASCIZ  /RK06 VEC ADR = /
4087  007450  041505  040440  051104
4088  007456  036440  000040
4089  007462  045522  033060  050040  RKPRTY: .ASCIZ  /RK06 PRIORITY =/
4090  007470  044522  051117  052111
4091  007476  020131  000075
4092  007502  053523  020122  020075  SWRMSG: .ASCIZ  /SWR = /
4093  007510  000000
4094  007511  000040  047040  053505  NEWMSG: .ASCIZ  /  NEW = /
4095  007516  036440  000040
4096  007522  005015  051104  053111  DRVSEQ: .ASCIZ  <15><12>/DRIVE(S) = /
4097  007530  024105  024523  036440
```

```
 4098  007536  000040
 4099  007540  051104  053111  020105  BADDRV: .ASCIZ  /DRIVE   /
 4100  007546  020040      000
 4101  007551     116  047117  042455  NXDRIV: .ASCIZ  /NON-EXISTENT/<15><12>
 4102  007556  044530  052123  047105
 4103  007564  006524  000012
 4104  007570  047516  020124  042522  NTREDY: .ASCIZ  /NOT READY/<15><12>
 4105  007576  042101  006531  000012
 4106  007604  051127  052111  020505  WRTLOK: .ASCIZ  /WRITE-LOCKED/<15><12>
 4107  007612  047514  045503  042105
 4108  007620  005015      000
 4109  007623     114  040517  042504  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
 4110  007630  020104  044527  044124
 4111  007636  040440  044514  047107
 4112  007644  050040  041501  006513
 4113  007652  000012
 4114  007654  042111  054040  042130  REPLPK: .ASCIZ  /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", &  REPLACE IT : /
 4115  007662  020120  040520  045503
 4116  007670  047440  020116  051104
 4117  007676  020126  026060  052040
 4118  007704  050131  020105  054440
 4119  007712  036040  051103  021076
 4120  007720  020054  020046  051040
 4121  007726  050105  040514  042503
 4122  007734  044440  020124  020072
 4123  007742      000
 4124  007743      015  005012  025052  NODRTS: .ASCII  <15><12><12>/** NO DRIVES TO TEST/<15><12>
 4125  007750  047040  020117  051104
 4126  007756  053111  051505  052040
 4127  007764  020117  042524  052123
 4128  007772  005015
 4129  007774  051120  051505  020123  CNTRDY: .ASCIZ  /PRESS "CONT" WHEN RDY/<15><12>
 4130  010002  041442  047117  021124
 4131  010010  053440  042510  020116
 4132  010016  042122  006531  000012
 4133  010024  040510  052114  051040  HLTRQD: .ASCIZ  /HALT REQUESTED/<15><12>
 4134  010032  050505  042525  052123
 4135  010040  042105  005015      000
 4136  010045     104  044522  042526  DRVXDP: .ASCIZ   DRIVE 0 IS LOAD MEDIUM/<15><12>
 4137  010052  030040  044440  020123
 4138  010060  047514  042101  046440
 4139  010066  042105  052511  006515
 4140  010074  000012
 4141  010076  005015  052012  020117  ALDRVS: .ASCIZ  <15><12><12>/TO TEST ALL DRIVES TYPE "A" <CR>, ELSE <CR><15> IE  +
 4142  010104  042524  052123  040440
 4143  010112  046114  042040  044522
 4144  010120  042526  020123  054524
 4145  010126  042520  021040  021101
 4146  010134  036040  051103  026076
 4147  010142  042440  051514  020105
 4148  010150  041474  037122  005015
 4149  010156  020052      000
 4150  010161      015  046012  036440  TSTMDS: .ASCII  <15><12>/L = LIST TESTS/<15><12>
 4151  010166  046040  051511  020124
 4152  010174  042524  052123  006523
 4153  010202      012
```

```
010203     103  036440  041440          .ASCII  /C = CHANGE TESTS/<15><12>
010210  043510  043516  020105
010216  042524  052123  006523
010224     012
010225     111  036440  044440          .ASCIZ  /I = INPUT PARAMS, RUN TESTS/<15><12>
010232  050116  052125  050040
010240  051101  046501  026123
010246  051040  047125  052040
010254  051505  051524  005015
010262     000
010263     015  042412  052116  ENTLCI: .ASCIZ  <15><12>/ENTER L,C, OR I/<15><12>/* /
010270  051105  046040  041454
010276  020054  051117  044440
010304  005015  020052     000
010311     124  020117  042504  DFTEST: .ASCIZ  /TO DEFAULT TESTS TYPE D <CR>, ELSE CR <15><12> *
010316  040506  046125  020124
010324  042524  052123  020123
010332  054524  042520  042040
010340  036040  051103  026076
010346  042440  051514  020105
010354  041474  037122  005015
010362  020052     000
010365     015  052012  051505  TLSTHC: .ASCIZ  <15><12>/TEST     ITERATIONS/<15><12>
010372  020124  020040  020040
010400  052111  051105  052101
010406  047511  051516  005015
010414     000
010415     015  052012  036440  EXPLAN: .ASCII  <15><12>/T = TYPE PARAM LIST/<15><12>
010422  052040  050131  020105
010430  040520  040522  020115
010436  044514  052123  005015
010444  020117  020075  050117          .ASCII  /O = OPEN PARAM LIST/<15><12>
010452  047105  050040  051101
010460  046501  046040  051511
010466  006524     012
010471     123  036440  051440          .ASCII  /S = SET INDIVIDUAL PARAM/<15><12>
010476  052105  044440  042116
010504  053111  042111  040525
010512  020114  040520  040522
010520  006515     012
010523     122  036440  051040          .ASCIZ  /R = RUN TESTS/<15><12>
010530  047125  052040  051505
010536  051524  005015     000
010543     015  042412  052116  PARMDE: .ASCIZ  <15><12>/ENTER T,O,S, OR R/<15><12>
010550  051105  052040  047454
010556  051454  020054  051117
010564  051040  005015     000
010571     015  025012  042040  DUACES: .ASCIZ  <15><12>/* DUAL-ACCESS DATA TEST */<15><12>
010576  040525  026514  041501
010604  042503  051523  042040
010612  052101  020101  042524
010620  052123  025040  005015
010626     000
010627     115  054101  053440  MAWRDC: .ASCIZ  /MAX WORD COUNT = /
010634  051117  020104  047503
010642  047125  020124  020075
```

```
 4210  010650     000
 4211  010651     052  020052  051440  SECNL1: .ASCII  /**  S0 S1/
 4212  010656  037060  030523
 4213  010662  047040  052117  042440  NOTPLC: .ASCIZ  / NOT ALLOWED/<15><12>
 4214  010670  046114  053517  042105
 4215  010676  005015     000
 4216  010701     052  020052  051440  SECNL2: .ASCIZ   **  S2 S3/
 4217  010706  037062  031523     000  TRKNLW: .ASCIZ  /**  FT LT/
 4218  010713     052  020052  043040
 4219  010720  037124  052114     000  WC2BIG: .ASCIZ  /**  WC OR MA TOO LARGE/<15><12>
 4220  010725     052  020052  053440
 4221  010732  020103  051117  046440
 4222  010740  020101  047524  020117
 4223  010746  040514  043522  006505
 4224  010754  000012
 4225  010756  047524  042040  043105  DFQUES: .ASCIZ  /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR>/<15><12>/* /
 4226  010764  052501  052114  040440
 4227  010772  046114  050040  051101
 4228  011000  046501  026123  052040
 4229  011006  050131  020105  020104
 4230  011014  041474  037122  020054
 4231  011022  046105  042523  036040
 4232  011030  051103  006476  025012
 4233  011036  000040
 4234  011040  005015  051525  051105  PFIFTN: .ASCIZ  <15><12>/USER-DEFINED PATTERN 15 :/<15><12>
 4235  011046  042055  043105  047111
 4236  011054  042105  050040  052101
 4237  011062  042524  047122  030440
 4238  011070  020065  006472  000012
 4239  011076  005015  047515  044504  SE_P15: .ASCIZ  <15><12>/MODIFY PATTERN 15 :/<15><12>
 4240  011104  054506  050040  052101
 4241  011112  042524  047122  030440
 4242  011120  020065  006472  000012
 4243  011126  047524  046440  042117  MDFY15: .ASCIZ  /TO MODIFY PATTERN 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
 4244  011134  043111  020131  040520
 4245  011142  052124  051105  020116
 4246  011150  032461  020054  054524
 4247  011156  042520  046440  036040
 4248  011164  051103  026076  042440
 4249  011172  051514  020105  041474
 4250  011200  037122  005015  020052
 4251  011206     000
 4252  011207     015  042412  052116  ENTPAS: .ASCIZ  <15><12>/ENTER NO. OF PASSES (1-77777) :/<15><12>/* /
 4253  011214  051105  047040  027117
 4254  011222  047440  020106  040520
 4255  011230  051523  051505  024040
 4256  011236  026461  033467  033467
 4257  011244  024467  035040  005015
 4258  011252  020052     000
 4259  011255     015  025012  020052  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRIVE - 20 ERRORS/<15><12>
 4260  011262  042040  047522  050120
 4261  011270  047111  020107  051104
 4262  011276  053111  020105  020055
 4263  011304  030062  042440  051122
 4264  011312  051117  006523  000012
 4265  011320  005015  052012  051505  TSTDRN: .ASCII  <15><12><12>/TESTING DRIVE /
```

```
 4266  011326  044524  043516  042040
 4267  011334  044522  042526     040
 4268  011341     040  005015     000  DRVNO:  .ASCIZ  / <15><12>
 4269  011346     104  044522  042526  DRIV:   .ASCIZ  /DRIVE
 4270  011352  000040
 4271  011354  040503  052122  020056  CART:   .ASCIZ  /CART. /
 4272  011362     000
 4273  011363     123  051105  020056  SERNM:  .ASCIZ  /SER. NO.  /
 4274  011370  047516  020056  000040
 4275  011376  005015  040506  052103  FACTBS: .ASCIZ  <15><12>/FACTORY /
 4276  011404  051117  020131     000
 4277  011411     012  047523  052106  SOFTBS: .ASCII  <12>/SOFTWARE /
 4278  011416  040527  042522     040
 4279  011423     102  042101  051440  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
 4280  011430  041505  047524  051522
 4281  011436  035040  005015     000
 4282  011443     040  047040  047117  NOFALS: .ASCIZ  /  NONE/
 4283  011450  000105
 4284
 4285                                  ;MESSAGES USED IN TIMING TESTS
 4286  011452  025052  020040  047516  NOCLKS: .ASCII  /**  NO CLOCK/
 4287  011460  041440  047514  045503
 4288  011466  026440  020040  044524  TIMSKP: .ASCIZ  / -  TIMING TESTS WILL BE SKIPPED/<15><12>
 4289  011474  044515  043516  052040
 4290  011502  051505  051524  053440
 4291  011510  046111  020114  042502
 4292  011516  051440  044513  050123
 4293  011524  042105  005015     000
 4294  011531     052  020052  045440  CLKFAL: .ASCIZ  /**  KW11 FAILURE/
 4295  011536  030527  020061  040506
 4296  011544  046111  051125  000105
 4297  011552  005015  047522  040524  ROTIMS: .ASCIZ  <15><12>/ROTATIONAL TIMES :/<15><12>
 4298  011560  044524  047117  046101
 4299  011566  052040  046511  051505
 4300  011574  035040  005015     000
 4301  011601     015  040412  042526  AVGSEK: .ASCIZ  <15><12>/AVERAGE SEEK TIMES :/<15><12>
 4302  011606  040522  042507  051440
 4303  011614  042505  020113  044524
 4304  011622  042515  020123  006472
 4305  011630  000012
 4306  011632  005015  047117  020105  ONECYL: .ASCIZ  <15><12>/ONE CYL SEEK TIMES :/<15><12>
 4307  011640  054503  020114  042523
 4308  011646  045505  052040  046511
 4309  011654  051505  035040  005015
 4310  011662     000
 4311  011663     015  046412  054101  MAXSEK: .ASCIZ  <15 <12>/MAXIMUM SEEK TIMES :/<15><12>
 4312  011670  046511  046525  051440
 4313  011676  042505  020113  044524
 4314  011704  042515  020123  006472
 4315  011712  000012
 4316  011714  025052  047506  053522  FORWRD: .ASCIZ  /**FORWARD DIRECTION**/<15><12>
 4317  011722  051101  020104  044504
 4318  011730  042522  052103  047511
 4319  011736  025116  006452  000012
 4320  011744  025052  042522  042526  REVRSE: .ASCIZ  /**REVERSE DIRECTION**/<15><12>
 4321  011752  051522  020105  044504
```

```
 4322  011760  042522  052103  047511
 4323  011766  025116  006452  000012
 4324  011774  044515  020116  020075  MINEQ:  .ASCIZ  /MIN = .
 4325  012002     000
 4326  012003     115  054101  036440  MAXEQ:  .ASCIZ  /MAX = /
 4327  012010  000040
 4328  012012  053101  020107  020075  AVGEQ:  .ASCIZ  /AVG = /
 4329  012020     000
 4330  012021     040  051525     000  MICROS: .ASCIZ  / US/
 4331  012025     040  043117  030440  BELOW:  .ASCIZ  / OF 128 BELOW SPEC'D MIN OF /
 4332  012032  034062  041040  046105
 4333  012040  053517  051440  042520
 4334  012046  023503  020104  044515
 4335  012054  020116  043117  000040
 4336  012062  047440  020106  031061  ABOVE:  .ASCIZ  . OF 128 ABOVE SPEC'D MAX OF /
 4337  012070  020070  041101  053117
 4338  012076  020105  050123  041505
 4339  012104  042047  046440  054101
 4340  012112  047440  020106     000
 4341  012117     040  043117  032040  ABOVE1: .ASCIZ    OF 410 ABOVE SPEC'D MAX OF 8000 US/<15><12>
 4342  012124  030061  040440  047502
 4343  012132  042526  051440  042520
 4344  012140  023503  020104  040515
 4345  012146  020130  043117  034040
 4346  012154  030060  020060  051525
 4347  012162  005015     000
 4348  012165     040  043117  030440  ABOVE3: .ASCIZ  / OF 128 ABOVE SPEC'D MAX OF 75000 US/<15><12>
 4349  012172  034062  040440  047502
 4350  012200  042526  051440  042520
 4351  012206  023503  020104  040515
 4352  012214  020130  043117  033440
 4353  012222  030065  030060  052440
 4354  012230  006523  000012
 4355  012234  020040  051440  042520  SPCDMX: .ASCIZ  /   SPEC'D MAX IS 38000 US/<15><12>
 4356  012242  023503  020104  040515
 4357  012250  020130  051511  031440
 4358  012256  030070  030060  052440
 4359  012264  006523  000012
 4360  012270  032062  033463  020065  LIM1:   .ASCIZ  /24375 US/<15><12>
 4361  012276  051525  005015     000
 4362  012303     062  033065  032462  LIM2:   .ASCIZ  /25625 US/<15><12>
 4363  012310  052440  006523  000012
 4364
 4365  012316  025052  020040  040503  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRACK!/<15><12>
 4366  012324  047116  052117  051040
 4367  012332  040505  020104  040502
 4368  012340  020104  042523  052103
 4369  012346  051117  052040  040522
 4370  012354  046503  006441  000012
 4371  012362  041536  005015     000  CNTRLC: .ASCIZ  /↑C/<15><12>
 4372  012367     136  006532  000012  CNTRLZ: .ASCIZ  /↑Z/<15><12>
 4373  012374  051136  005015     000  CNTRLR: .ASCIZ  /↑R/<15><12>
 4374  012401     136  006525  000012  CNTRLU: .ASCIZ  /↑U/<15><12>
 4375  012406  043536  005015     000  CNTRLG: .ASCIZ  /↑G/<15><12>
 4376  012413     015  005012     000  CR2LF:  .ASCIZ  <15><12><12>
 4377  012417     134     000          BKSLSH: .ASCIZ  /\/
```

```
012421     054    000                   COMMA:   .ASCIZ  /,/
012423     040    040                   SPACE6:  .ASCII  /  /
012425     040                          SPACE4:  .ASCII  /  /
012426     040                          SPACE3:  .ASCII  /  /
012427     040                          SPACE2:  .ASCII  / /
012430  000040                          SPACE1:  .ASCIZ  / /
                                                 .EVEN
012432  020040  000075                  PRMBUF:  .ASCIZ  /  =/
012436  047527  042122  000040          WORDSP:  .ASCIZ  /WORD /
012444  036440  000040                  EQUALS:  .ASCIZ  / = /
012450  020052     000                  PROMPT:  .ASCIZ  /* /
012453     076  000040                  PRMPSP:  .ASCIZ  /> /

                                                 .EVEN




012456  105037  003106                  DFSTRT:  CLRB    MDFLAG          ;SET FLAG FOR DEFAULT MODE
012462  105037  003126                           CLRB    DULACS          ;CLEAR DUAL-ACCESS FLAG
012466  105037  003116                           CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
012472  022737  013660  000042                   CMP     #DRVTST,@#42    ;SEE IF EOP RETURN ADRS = DRVTST
012500  001003                                   BNE     4$              ;BR IF NOT DRVTST
012502  012737  016760  000042                   MOV     #NEWPAS,@#42    ;SET RETURN ADRS = NEWPAS
012510  000414                          4$:      BR      CMSTRT          ;PROCEED

012512  112737  000001  003126          DASTRT:  MOVB    #1,DULACS       ;SET FLAG FOR DUAL-ACCESS DATA TEST
012520  112737  000001  003106                   MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
012526  000405                                   BR      CMSTRT          ;PROCEED

012530  112737  000001  003106          PSTART:  MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
012536  105037  003126                           CLRB    DULACS          ;CLEAR DUAL-ACCESS TEST FLAG

012542  012737  000340  177776          CMSTRT:  MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
                                        .SBTTL   INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
012550  012706  001100                           MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
012554  005026                                   CLR     (R6)+           ;;CLEAR MEMORY LOCATION
012556  022706  001140                           CMP     #SWR,R6 ;;DONE?
012562  001374                                   BNE     .-6             ;;LOOP BACK IF NO
012564  012706  001100                           MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
012570  012737  054372  000020                   MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
012576  012737  000340  000022                   MOV     #340,@#IOTVEC+2 ;;LEVEL 7
012604  012737  053666  000030                   MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
012612  012737  000340  000032                   MOV     #340,@#EMTVEC+2 ;;LEVEL 7
012620  012737  055500  000034                   MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
012626  012737  000340  000036                   MOV     #340,@#TRAPVEC+2;LEVEL 7
012634  012737  054236  000024                   MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
012642  012737  000340  000026                   MOV     #340,@#PWRVEC+2 ;;LEVEL 7
012650  013737  025024  025016                   MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
012656  005037  001304                           CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
012662  005037  001306                           CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
012666  112737  000001  001115                   MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
012674  012737  012674  001106                   MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
012702  012737  012702  001110                   MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
```

```
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
012710  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
012714  012737  012750  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
012722  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
012730  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
012736  022777  177777  166174                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
012744  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
012746  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
012750  012716  012756                64$:      MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
012754  000002                                  RTI
012756  012737  000176  001140        65$:      MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
012764  012737  000174  001142                  MOV     #DISPREG,DISPLAY
012772  012637  000004                66$:      MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

012776  005037  001326                          CLR     $PASS           ;;CLEAR PASS COUNT
013002  132737  000200  001341                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
013010  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
013012  012737  001342  001140                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
013020                                67$:
013020  000005                                  RESET                   ;CLEAR THE UNIBUS
013022  012737  000006  000004                  MOV     #6,@#ERRVEC     ;SET TIME-OUT VECTOR
013030  005037  000006                          CLR     @#ERRVEC+2
013034  012737  026142  000114                  MOV     #MPERHD,@#MEMVEC ;SET MEM PARITY TRAP VECTOR
013042  012737  000340  000116                  MOV     #PR7,@#MEMVEC+2 ;SET TRAP PRIORITY = 7
013050  004737  026056                          JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
013054  112737  000144  001115                  MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
013062  105037  003120                          CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
013066  112737  000001  003136                  MOVB    #1,HLPOVL       ;SET HLP FILE OVLD INDICTR
013074  104401  007176                          TYPE    .DZR6M          ;TYPE PROGRAM I.D. FOR PART 1
013100  104401  007216                          TYPE    ,SUBVER
013104  104401  007273                          TYPE    ,PART1
013110  105737  003137                          TSTB    NOTYPE          ;SEE IF OPERATOR NOTE SHOULD BE TYPED
013114  001004                                  BNE     39$             ;BR IF NOT
013116  104401  063526                          TYPE    ,NOTMSG         ;TYPE OPERATOR NOTE
013122  105237  003137                          INCB    NOTYPE          ;SET FLAG FOR NEXT TIME
013126  105737  003126                39$:      TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
013132  001402                                  BEQ     40$             ;BR IF NOT
013134  104401  010571                          TYPE    ,DUACES         ;TYPE "* DUAL-ACCESS DATA TEST *"
013140  012737  025170  000060        40$:      MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
013146  012737  000200  000062                  MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
013154  013701  003030                          MOV     RKVEC,R1        ;ADDR. OF RK06 VECTOR STORAGE
013160  012721  044506                          MOV     #I.INTR,(R1)+   ;SET IT TO RK06 HANDLER
013164  013711  003032                          MOV     RKPRI,(R1)      ;SET RK06 PRIORITY
013170  012737  025762  000250                  MOV     #KTERHD,@#MMVEC ;VECT FOR KT11 FAILURE
013176  012737  000340  000252                  MOV     #PR7,@#MMVEC+2  ;SET PRIOR. = 7 FOR HNDLER
013204  105037  003110                          CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
013210  012737  000000  177776                  MOV     #PRO,@#PS       ;ALLOW ALL INTERRUPTS AGAIN
013216  012701  005222                          MOV     #PRVCMD,R1      ;ZERO OUT PREVIOUS COMMAND
013222  012700  000006                          MOV     #6,R0           ;SET COUNTER
013226  005021                        42$:      CLR     (R1)+           ;ZERO A WORD
013230  005300                                  DEC     R0
013232  001375                                  BNE     42$             ;BR IF NOT DONE YET
013234  005037  005502                          CLR     STALLS          ;DON'T ALLOW STALLS YET
013240  004737  025642                          JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
```

```
4490  013244  012737  176543  052134  44$:    MOV     #176543,$HINUM  ;INIT. PSEUDO-RANDOM NOS.
4491  013252  012737  123456  052136          MOV     #123456,$LONUM
4492                                  ;CHECK FOR PRESENCE OF KW11-L OR P CLOCK, AND SET FLAGS
4493  013260  105037  003122                  CLRB    PCLKF           ;INIT. P CLOCK FLAG
4494  013264  105037  003123                  CLRB    DOTIM           ;INIT. TIMING TESTS FLAG
4495  013270  012737  013330  000004          MOV     #6$,@#4         ;SET TIME-OUT ERROR VECTOR
4496  013276  005777  167640                  TST     @PKS            ;SEE IF P-CLOCK IS PRESENT
4497  013302  105237  003122                  INCB    PCLKF           ;PRESENT, SET FLAG
4498  013306  013700  003152                  MOV     PCVEC,R0        ;LOAD KW11-P VECTOR ADDRESS
4499  013312  012720  032232          4$:     MOV     #CLOCK,(R0)+    ;ADDR OF CLOCK SERVICE ROUTINE
4500  013316  012710  000340                  MOV     #PR7,(R0)       ;SET CLOCK HANDLER PRIORITY = 7
4501  013322  105237  003123                  INCB    DOTIM           ;SET FLAG TO ALLOW TIMING TESTS
4502  013326  000414                          BR      8$              ;BR TO CONTINUE
4503  013330  022626                  6$:     CMP     (SP)+,(SP)+     ;P-CLK NOT THERE, RESET THE STACK
4504  013332  012737  013352  000004          MOV     #7$,@#4         ;SET TIME-OUT ERROR VECTOR
4505  013340  005777  167574                  TST     @LKS            ;SEE IF L-CLK PRESENT
4506  013344  013700  003150                  MOV     LCVEC,R0        ;LOAD KW11-L VECTOR ADDRESS
4507  013350  000760                          BR      4$              ;BR TO SET UP L-CLK VECTOR
4508  013352  022626                  7$:     CMP     (SP)+,(SP)+     ;L-CLK NOT THERE, RESET THE STACK
4509  013354  104401  011452                  TYPE    ,NOCLKS         ;SAY TIMING TESTS WON'T BE RUN
4510  013360  012737  000006  000004  8$:     MOV     #6,@#4          ;SET TIME-OUT VECTOR TO LOCATION 6
4511  013366  004737  025362                  JSR     PC,SIZMEM       ;SIZE MEMORY, FIX MA LIMIT IN PRMLIM
4512  013372  105737  003125                  TSTB    XDPSVD          ;SEE IF XXDP PREVIOUSLY SAVED
4513  013376  001404                          BEQ     9$              ;BR IF NOT
4514  013400  004737  027040                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP
4515  013404  105037  003125                  CLRB    XDPSVD          ;CLEAR THE FLAG
4516  013410  105737  003106          9$:     TSTB    MDFLAG          ;SEE IF DEFAULT MODE
4517  013414  001031                          BNE     10$             ;BR IF NOT DEFAULT MODE
4518  013416  013700  001370                  MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
4519  013422  013737  001374  003026          MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
4520  013430  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
4521  013436  001005                          BNE     18$             ;BR IF NO SIZING
4522  013440  012700  120210                  MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
4523  013444  012737  177440  003026          MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
4524  013452  110037  003030          18$:    MOVB    R0,RKVEC        ;STORE VECTOR
4525  013456  105037  003031                  CLRB    RKVEC+1         ;CLEAR HI BYTE
4526  013462  000300                          SWAB    R0              ;GET PRTY INTO BITS 5-7
4527  013464  042700  177437                  BIC     #177437,R0      ;CLEAR OTHER BITS
4528  013470  010037  003032                  MOV     R0,RKPRI        ;STORE RK06 PRIORITY
4529  013474  000137  014376                  JMP     ALLDRV          ;GO CHECK ALL DRIVES
4530
4531
4532
4533                                  ;BEGIN PARAMETER INPUT MODE
4534  013500  104401  007370          10$:    TYPE    ,PRMINP         ;TYPE "PARAMETER INPUT MODE"
4535
4536                                  ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
4537  013504  013746  003026                  MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
4538  013510  104401  007422                  TYPE    ,RKBADR         ;TYPE "RK06 BUS ADR = "
4539  013514  004737  025532                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
4540  013520  012637  003026                  MOV     (SP)+,RKBAS     ;STORE NEW VALUE
4541                                  ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
4542  013524  013746  003030                  MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
4543  013530  104401  007442                  TYPE    ,RKVADR         ;TYPE "RK06 VEC ADR = "
4544  013534  004737  025532                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
4545  013540  012637  003030                  MOV     (SP)+,RKVEC     ;STORE NEW VALUE
```

```
4546
4547  013544  013746  003032          11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
4548  013550  006316                          ASL     (SP)            ;GET IT INTO BITS 0-2
4549  013552  006316                          ASL     (SP)
4550  013554  006316                          ASL     (SP)
4551  013556  000316                          SWAB    (SP)
4552  013560  104401  007462                  TYPE    ,RKPRTY         ;TYPE   "RK06 PRIORITY = "
4553  013564  004737  025532                  JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
4554  013570  012600                          MOV     (SP)+,R0
4555  013572  020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
4556  013576  002414                          BLT     12$             ;BR IF NEW VALUE TOO SMALL
4557  013600  020027  000007                  CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
4558  013604  003011                          BGT     12$             ;BR IF NEW VALUE TOO LARGE
4559  013606  000300                          SWAB    R0              ;GET PRIORITY INTO BITS 5-7
4560  013610  006200                          ASR     R0
4561  013612  006200                          ASR     R0
4562  013614  006200                          ASR     R0
4563  013616  010037  003032                  MOV     R0,RKPRI        ;STORE NEW VALUE
4564  013622  104401  001315                  TYPE    ,$CRLF
4565  013626  000405                          BR      16$
4566  013630  104401  005262          12$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
4567  013634  104401  001314                  TYPE    ,$QUES
4568  013640  000741                          BR      11$             ;GO ASK AGAIN
4569
4570  013642  105037  003116          16$:    CLRB    ERRCNT          ;CLEAR ERROR CNT FOR RESTARTS
4571  013646  012737  013660  000042          MOV     #DRVTST,@#42    ;SET UP DUMP MODE RETURN FROM $EOP
4572                                                                  ;    UPON COMPLETION OF REQUESTED PASSES
4573  013654  000137  014470                  JMP     CHKLST          ;GO CHECK STATUS OF MARKED DRIVES
4574
4575
4576
4577                                  ;;*****************************************************************
4578                                  .SBTTL  TO - DESIRED DRIVE INPUT ROUTINE
4579                                  ;*THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
4580                                  ;*WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
4581                                  ;*CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
4582                                  ;*DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
4583                                  ;*AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
4584                                  ;*LIST (DRVLST).
4585                                  ;;*****************************************************************
4586
4587  013660                          DRVTST:
4588  013660  012706  001100                  MOV.    #STACK,SP       ;RESET THE STACK
4589  013664  005037  005502                  CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
4590  013670  004737  026056                  JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
4591  013674  104401  010076                  TYPE    ,ALDRVS         ;ASK IF ALL DRIVES DESIRED
4592  013700  004737  030174                  JSR     PC,RDCHRS       ;READ RESPONSE
4593  013704  013660                          DRVTST                  ;(↑C) RETURN ADDRESS
4594  013706  013660                          DRVTST                  ;(↑Z) RETURN ADDRESS
4595  013710  013660                          DRVTST                  ;(↑U) OR ERROR RETURN ADDRESS
4596  013712  005700                          TST     R0              ;SEE IF NULL INPUT
4597  013714  001413                          BEQ     TELDRV          ;BR IF NULL INPUT
4598  013716  022737  000101  005262          CMP     #'A,BUFF0       ;SEE IF ALL DRIVES REQUESTED
4599  013724  001002                          BNE     4$              ;BR IF NOT ALL DRIVES
4600  013726  000137  014376                  JMP     ALLDRV          ;CHECK ALL DRIVES
4601  013732  104401  005262          4$:     TYPE    ,BUFF0          ;ECHO BAD INPUT
```

```
                        TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
                        BR      DRVTST          ;GO ASK AGAIN
                ;TYPE CURRENT DRIVE LIST
                TELDRV: TYPE    ,DRVSEQ         ;TYPE "DRIVE(S) = "
                        CLR     R1              ;INITIALIZE DRIVE NUMBER
                        CLR     R0              ;INIT. COUNT OF LISTED DRIVES
                        MOV     #BIT0,R3        ;INIT BIT POINTER
                4$:     TSTB    DRVLST(R1)      ;SEE IF THIS DRIVE IS LISTED
                        BEQ     8$              ;BR IF DRIVE NOT LISTED
                        TST     R0              ;SEE IF FIRST TIME HERE
                        BEQ     6$              ;BR IF FIRST TIME HERE
                        TYPE    ,COMMA          ;TYPE A COMMA
                6$:     MOV     R1,SCRACH       ;USE SCRACH FOR BUFFER
                        BIS     #'0,SCRACH      ;CONVERT DRIVE NO. TO ASCII
                        TYPE    ,SCRACH         ;TYPE DRIVE NO.
                        INC     R0              ;INCREMENT NO. OF LISTED DRIVES
                8$:     INC     R1              ;INCREMENT DRIVE NO.
                        ASL     R3              ;SHIFT BIT POINTER
                        CMP     #10,R1          ;SEE IF DONE YET
                        BNE     4$              ;BR IF NOT DONE
                        TST     R0              ;SEE IF ANY DRIVES WERE LISTED
                        BNE     26$             ;BR IF YES
                        TYPE    ,NOFALS         ;TYPE "  NONE"
                        TYPE    ,NODRTS         ;TYPE "** NO DRIVES TO TEST"
                                                ;     "PRESS 'CONT' WHEN RDY"
                        MOV     #401,R3         ;PATTERN TO MARK DRIVES
                        MOV     #DRVLST,R1      ;DRIVE LIST ADDRESS
                        MOV     R3,(R1)+        ;MARK ALL DRIVES IN LIST
                        MOV     R3,(R1)+
                        MOV     R3,(R1)+
                        MOV     R3,(R1)
                        JMP     HLTPRG          ;HALT PROGRAM
                26$:    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
                        TSTB    MDFLAG          ;SEE IF DEFAULT MODE
                        BNE     20$             ;BR IF NOT DEFAULT MODE
                        JMP     LODFLS          ;GO LOAD DEFLT ITER. COUNTS
                20$:    CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
                        BNE     19$             ;BR IF NOT
                        TSTB    DRVLST          ;SEE IF DRIVE 0 LISTED TO TEST
                        BEQ     19$             ;BR IF NOT
                24$:    TYPE    ,REPLPK         ;TYPE "IF XXDP PACK ON DRV 0, TYPE Y <CR>
                                                ;     AND REPLACE IT"
                        JSR     PC,RDCHRS       ;READ RESPONSE
                        24$                     ;(↑C) RETURN
                        24$                     ;(↑Z) RETURN
                        24$                     ;(↑U) RETURN
                        TST     R0              ;SEE IF NULL INPUT
                        BEQ     19$             ;BR IF JUST <CR> TYPED
                        CMP     #'Y,BUFF0       ;SEE IF "Y <CR>" TYPED
                        BNE     24$             ;BR IF NOT, TO ASK AGAIN
                        TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN DRV RDY"
                        BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
                        HALT                    ;HALT FOR PACK CHANGE
                        JSR     PC,INITSS       ;INIT THE SUB-SYS
                        JMP     CHKLST          ;GO CHECK STATUS OF LISTED DRIVES
                19$:    TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
```

```
                                        ;READ AND CHECK NEW DRIVE NUMBERS
014206  004737  030174                      JSR     PC,RDCHRS       ;READ IN INPUT STRING
014212  013660                              DRVTST                  ;(↑C) RETURN ADDRESS
014214  013660                              DRVTST                  ;(↑Z) RETURN ADDRESS
014216  013744                              TELDRV                  ;(↑U) OR ERROR RETURN ADDRESS
014220  005700                              TST     R0              ;SEE IF NULL INPUT
014222  001007                              BNE     9$              ;BR IF NEW DRIVES WILL BE SELECTED
014224  105737  003126                      TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
014230  001002                              BNE     22$             ;BR IF YES
014232  000137  014534                      JMP     ASKTST          ;JUMP TO THE TEST INPUT ROUTINE
014236  000137  015300          22$:        JMP     INPUTP          ;GO ASK FOR INPUT PARAMETERS
014242  022700  000017          9$:         CMP     #15.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
014246  002005                              BGE     12$             ;BR IF NOT TOO MANY
014250  104401  005262          10$:        TYPE    ,BUFFO          ;ECHO BAD INPUT
014254  104401  001314                      TYPE    ,$QUES          ;TYPE <?> AND <CR><LF>
014260  000631                              BR      TELDRV          ;GO TYPE DRIVES AND ASK AGAIN
014262  005001                  12$:        CLR     R1              ;CLEAR CHAR. POINTER
014264  126127  005262  000060  14$:        CMPB    BUFFO(R1),#'0   ;SEE IF DRIVE NO. < 0
014272  002766                              BLT     10$             ;BR TO ECHO BAD INPUT
014274  126127  005262  000067              CMPB    BUFFO(R1),#'7   ;SEE IF > 7
014302  003362                              BGT     10$             ;BR TO ECHO BAD INPUT
014304  005201                              INC     R1              ;INCREMENT CHAR POINTER
014306  020100                              CMP     R1,R0           ;SEE IF MORE CHARS TO CHECK
014310  001406                              BEQ     16$             ;BR IF ALL CHARS CHECKED
014312  126127  005262  000054              CMPB    BUFFO(R1),#',   ;SEE IF THIS IS COMMA
014320  001353                              BNE     10$             ;BR IF NOT COMMA
014322  005201                              INC     R1              ;INCREMENT CHAR POINTER
014324  000757                              BR      14$             ;BR TO CONTINUE CHECKING CHARS
                                        ;GET NEW DRIVE NUMBERS INTO LIST
014326  012701  005576          16$:        MOV     #DRVLST,R1      ;GET DRIVE LIST ADDRESS
014332  005021                              CLR     (R1)+           ;CLEAR OUT THE DRIVE LIST
014334  005021                              CLR     (R1)+
014336  005021                              CLR     (R1)+
014340  005011                              CLR     (R1)
014342  005000                              CLR     R0              ;CLEAR BUFFER POINTER
014344  116001  005262          18$:        MOVB    BUFFO(R0),R1    ;GET A DRIVE NUMBER
014350  042701  000060                      BIC     #'0,R1          ;STRIP ASCII BITS
014354  112761  000001  005576              MOVB    #1,DRVLST(R1)   ;MARK THIS DRIVE IN DRIVE LIST
014362  062700  000002                      ADD     #2,R0           ;POINT TO NEXT DRIVE NUMBER
014366  105760  005261                      TSTB    BUFFO-1(R0)     ;SEE IF NULL CHAR YET
014372  001364                              BNE     18$             ;BR IF NOT DONE YET
014374  000435                              BR      CHKLST          ;GO CHECK NEW DRIVES FOR VALID STATUS
                                        ;COME HERE IF ALL DRIVES REQUESTED
014376  005000                  ALLDRV:     CLR     R0              ;CLEAR DRIVE NUMBER
014400  013703  001376                      MOV     $DEVM,R3        ;GET APT DEVICE MAP
014404  132737  000200  001341              BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
014412  001002                              BNE     1$              ;BR IF NO SIZING
014414  012703  000377                      MOV     #377,R3         ;SET UP FOR SIZING
014420  012701  000001          1$:         MOV     #BIT0,R1        ;SET BIT POINTER
014424  105060  005576          2$:         CLRB    DRVLST(R0)      ;INITIALIZE DRIVE ENTRY
014430  030103                              BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
014432  001403                              BEQ     4$              ;BR IF NOT
014434  112760  000001  005576              MOVB    #1,DRVLST(R0)   ;MARK THIS DRIVE IN DRIVE LIST
014442  006301                  4$:         ASL     R1              ;SHIFT BIT POINTER
014444  005200                              INC     R0              ;INCREMENT DRIVE NUMBER
014446  022700  000010                      CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
```

```
 4714  014452  001364                          BNE     2$              ;BR IF NOT DONE YET
 4715  014454  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF PROGRAM SHOULD SIZE
 4716  014462  001402                          BEQ     CHKLST          ;BR IF YES
 4717  014464  000137  013744                  JMP     TELDRV          ;GO LIST APT-SELECTED DRIVES
 4718                          ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
 4719  014470  005000          CHKLST: CLR     R0              ;CLEAR DRIVE NUMBER
 4720  014472  105760  005576  2$:     TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
 4721  014476  001405                          BEQ     4$              ;BR IF NOT MARKED
 4722  014500  010037  005500                  MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
 4723  014504  004737  027420                  JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE
 4724                                                          ;IF NOT VALID, TYPE MESSAGE
 4725                                                          ;   AND REMOVE IT FROM DRIVE LIST
 4726  014510  014526                          6$                      ;ERROR RETURN ADDRESS FOR SCNDRV
 4727  014512  005200          4$:     INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
 4728  014514  022700  000010                  CMP     #10,R0          ;SEE IF DONE CHECKING LIST
 4729  014520  001364                          BNE     2$              ;BR IF NOT DONE YET
 4730  014522  000137  013744                  JMP     TELDRV          ;GO BACK TO LIST SELECTED DRIVES
 4731  014526  105060  005576  6$:     CLRB    DRVLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
 4732  014532  000767                          BR      4$              ;CONTINUE CHECKING THE LIST
 4733
 4734
 4735                          ;;*****************************************************************
 4736                          .SBTTL  TO - DESIRED TEST INPUT ROUTINE
 4737                          ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
 4738                          ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY.
 4739                          ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
 4740                          ;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
 4741                          ;*THAT THE TEST IS NOT TO BE RUN.
 4742                          ;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
 4743                          ;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
 4744                          ;*TYPED, (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
 4745                          ;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
 4746                          ;*OF OPERATING PARAMETERS, AND RUN TESTS.
 4747                          ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
 4748                          ;* (↑Z) CAUSES RETURN TO ASKTMD.
 4749                          ;;*****************************************************************
 4750
 4751                          ;DETERMINE L,C, OR I MODE
 4752  014534  104401  010161  ASKTST:  TYPE    ,TSTMDS         ;ASK FOR TEST INPUT MODE
 4753  014540  104401  010263  ASKTMD:  TYPE    ,ENTLCI         ;TYPE "ENTER L,C, OR I"
 4754  014544  004737  030174          JSR     PC,RDCHRS       ;READ RESPONSE
 4755  014550  013660                  DRVTST                  ;(↑C) RETURN ADDRESS
 4756  014552  014540                  ASKTMD                  ;(↑Z) RETURN ADDRESS
 4757  014554  014540                  ASKTMD                  ;(↑U) OR ERROR RETURN ADDRESS
 4758  014556  005700                  TST     R0              ;SEE IF ANY INPUT
 4759  014560  001005                  BNE     4$              ;BR IF ANY INPUT
 4760  014562  104401  005262  2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
 4761  014566  104401  001314          TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 4762  014572  000762                  BR      ASKTMD          ;GO ASK AGAIN
 4763  014574  022737  000114  005262  4$:     CMP     #'L,BUFFO       ;SEE IF (L) TYPED
 4764  014602  001002                  BNE     6$              ;BR IF NOT (L)
 4765  014604  000137  014640          JMP     LSTTST          ;JUMP TO LIST TESTS
 4766  014610  022737  000103  005262  6$:     CMP     #'C,BUFFO       ;SEE IF (C) TYPED
 4767  014616  001002                  BNE     8$              ;BR IF NOT (C)
 4768  014620  000137  014756          JMP     CHGTST          ;JUMP TO CHANGE TESTS
 4769  014624  022737  000111  005262  8$:     CMP     #'I,BUFFO       ;SEE IF (I) TYPED
```

```
014632  001353                            BNE     2$                 ;BR IF NOT (I), TO ECHO BAD INPUT
014634  000137  015300                    JMP     INPUTP             ;GO INPUT PARAMS AND RUN TESTS

                          ;LIST CURRENT TESTS AND ITERATION COUNTS
014640  104401  010365    LSTTST: TYPE    ,TLSTHD            ;TYPE HEADING "TEST     ITERATIONS"
014644  005001                    CLR     R1                 ;INITIALIZE TEST INDEX
014646  004737  025330    2$:     JSR     PC,PREPKB          ;PREPARE FOR POSSIBLE KBD INPUT
014652  004737  030506            JSR     PC,TYPTST          ;TYPE CURRENT TEST AND ITERATION NUMBER
014656  104401  001315            TYPE    ,$CRLF             ;TYPE <CR>,<LF>
014662  005737  005522            TST     INTCHR             ;SEE IF ANY INPUT
014666  001416                    BEQ     8$                 ;BR IF NO INPUT
014670  122737  000003  005522    CMPB    #003,INTCHR        ;SEE IF (↑C) TYPED
014676  001002                    BNE     4$                 ;BR IF NOT (↑C)
014700  000137  013660            JMP     DRVTST             ;JUMP TO ASK FOR DRIVES AGAIN
014704  122737  000032  005522 4$: CMPB   #032,INTCHR        ;SEE IF (↑Z) TYPED
014712  001002                    BNE     6$                 ;BR IF NOT (↑Z)
014714  000137  014540            JMP     ASKTMD             ;JUMP TO ASK FOR NEW TEST INPUT MODE
014720  004737  025350    6$:     JSR     PC,ECOBAD          ;ECHO BAD INPUT
014724  062701  000002    8$:     ADD     #2,R1              ;INCREMENT TEST INDEX
014730  010102                    MOV     R1,R2              ;GET COPY OF INDEX
014732  062702  005606            ADD     #TSTLST,R2         ;GET POSITION IN LIST
014736  022702  005650            CMP     #DFLTST,R2         ;SEE IF DONE WITH LIST
014742  001341                    BNE     2$                 ;BR IF NOT DONE YET
014744  042777  000100  164172    BIC     #BIT6,@STKS        ;DISABLE KBD INTERRUPT
014752  000137  014540            JMP     ASKTMD             ;GO ASK FOR NEW TEST INPUT MODE

                          ;CHANGE CURRENT TESTS AND ITERATION COUNTS
014756  104401  010311    CHGTST: TYPE    ,DFTEST            ;ASK IF TESTS SHOULD BE DEFAULTED
014762  004737  030174            JSR     PC,RDCHRS          ;READ RESPONSE
014766  013660                    DRVTST                     ;(↑C) RETURN ADDRESS
014770  014540                    ASKTMD                     ;(↑Z) RETURN ADDRESS
014772  014756                    CHGTST                     ;(↑U) OR ERROR RETURN ADDRESS
014774  005700                    TST     R0                 ;SEE IF NULL INPUT
014776  001426                    BEQ     NULINP             ;BR IF NULL INPUT
015000  022737  000104  005262    CMP     #'D,BUFFO          ;SEE IF (D) TYPED
015006  001405                    BEQ     LODFLS             ;BR IF DEFAULTS REQUESTED
015010  104401  005262            TYPE    ,BUFFO             ;ECHO BAD INPUT
015014  104401  001314            TYPE    ,$QUES             ;TYPE (?) AND <CR>,<LF>
015020  000756                    BR      CHGTST             ;GO ASK AGAIN
                          ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
015022  012700  005650    LODFLS: MOV     #DFLTST,R0         ;LOAD DEFAULT LIST ADDRESS
015026  012702  005606            MOV     #TSTLST,R2         ;LOAD TEST LIST ADDRESS
015032  012022            4$:     MOV     (R0)+,(R2)+        ;LOAD A DEFAULT VALUE
015034  022702  005650            CMP     #DFLTST,R2         ;SEE IF DONE YET
015040  001374                    BNE     4$                 ;BR IF NOT DONE YET
015042  105737  003106            TSTB    MDFLAG             ;SEE IF DEFAULT MODE
015046  001234                    BNE     ASKTMD             ;BR IF NOT DEFAULT MODE
015050  000137  016210            JMP     LODFPT             ;GO LOAD DEFAULT PARAMETERS
015054  005001            NULINP: CLR     R1                 ;INITIALIZE TEST INDEX
015056  104401  010365            TYPE    ,TLSTHD            ;TYPE HEADING "TEST     ITERATIONS"
                          ;TYPE CURRENT TEST AND ITERATION NUMBER
015062  004737  030506    8$:     JSR     PC,TYPTST          ;TYPE A TEST NO. AND ITERATION NO.
015066  104401  012430            TYPE    ,SPACE1            ;TYPE A SPACE
015072  104401  012450            TYPE    ,PROMPT            ;TYPE ASTERISK AND SPACE
                          ;READ AND CHECK INPUT, IF ANY
015076  004737  030174            JSR     PC,RDCHRS          ;READ OCTAL DIGITS TYPED
```

```
015102  013660                                  DRVTST                  ;(†C) RETURN ADDRESS
015104  014540                                  ASKTMD                  ;(†Z) RETURN ADDRESS
015106  015062                                  8$                      ;(†U) OR ERROR RETURN ADDRESS
015110  005700                                  TST     R0              ;SEE IF ANY INPUT
015112  001012                                  BNE     12$             ;BR IF ANY INPUT
015114  062701  000002                  10$:    ADD     #2,R1           ;INCREMENT INDEX
015120  010102                                  MOV     R1,R2           ;COPY INDEX
015122  062702  005606                          ADD     #TSTLST,R2      ;GET POSITION IN LIST
015126  022702  005650                          CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
015132  001353                                  BNE     8$              ;BR IF NOT DONE YET
015134  000137  014540                          JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
015140  022737  000041  005262          12$:    CMP     #'!,BUFFO       ;SEE IF (!) TYPED
015146  001431                                  BEQ     16$             ;BR TO PROPAGATE CURRENT ITER. NO.
015150  005002                                  CLR     R2              ;INITIALIZE (!) INDICATOR
015152  122760  000041  005261                  CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
015160  001004                                  BNE     14$             ;BR IF NOT (!)
015162  105060  005261                          CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
015166  005300                                  DEC     R0              ;DECREMENT CHAR COUNT
015170  005202                                  INC     R2              ;SET (!) INDICATOR
015172  022700  000005                  14$:    CMP     #5,R0           ;SEE HOW MANY CHARS NOW
015176  002433                                  BLT     20$             ;BR IF TOO MANY (MAX ITER. NO. = 77776)
015200  012746  005262                          MOV     #BUFFO,-(SP)    ;GET BUF ADDR. ON STACK FOR OCTBIN
015204  004737  051702                          JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
015210  015266                                  20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
015212  012600                                  MOV     (SP)+,R0        ;GET ITERATION NUMBER
015214  020027  077776                          CMP     R0,#77776       ;SEE IF > 77776
015220  101022                                  BHI     20$             ;BR IF TOO BIG
015222  010061  005606                          MOV     R0,TSTLST(R1)   ;PUT ITERATION NUMBER INTO TEST LIST
015226  005702                                  TST     R2              ;SEE IF (!) WAS TYPED
015230  001731                                  BEQ     10$             ;BR IF NOT (!)
                                        ;PROPAGATE CURRENT ITERATION NO. TO THE END OF TSTLST
015232  010102                          16$:    MOV     R1,R2           ;COPY INDEX
015234  062702  005606                          ADD     #TSTLST,R2      ;GET POSITION IN LIST
015240  022702  005646                          CMP     #DFLTST-2,R2    ;SEE IF AT END OF LIST
015244  001002                                  BNE     17$             ;BR IF NOT DONE
015246  000137  014540                          JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
015252  016161  005606  005610          17$:    MOV     TSTLST(R1),TSTLST+2(R1) ;PROPAGATE TO NEXT WORD
015260  062701  000002                          ADD     #2,R1           ;INCREMENT INDEX
015264  000762                                  BR      16$             ;BR TO CONTINUE
015266  104401  005262                  20$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
015272  104401  001314                          TYPE    ,$QUES          ;TYPE (?) AND (CR),(LF
015276  000671                                  BR      8$              ;GO ASK AGAIN

                                        ;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
015300  104401  010415                  INPUTP: TYPE    ,EXPLAN         ;EXPLAIN INPUT MODES
015304  005037  005502                  ASKMDE: CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
015310  004737  026056                          JSR     PC,ENBCSR       ;ENABLE MEMORY PARITY CHECK
015314  104401  010543                          TYPE    ,PARMDE         ;ASK FOR DESIRED INPUT MODE
015320  104401  012456                          TYPE    ,PROMPT         ;TYPE "* "
015324  004737  030174                          JSR     PC,RDCHRS       ;READ RESPONSE TO MODE QUESTION
015330  013660                                  DRVTST                  ;(†C) RETURN ADDRESS
015332  015304                                  ASKMDE                  ;(†Z) RETURN ADDRESS
015334  015304                                  ASKMDE                  ;(†U) OR ERROR RETURN ADDRESS
015336  005700                                  TST     R0              ;SEE IF NULL INPUT
015340  001005                                  BNE     4$              ;BR IF ANY INPUT
015342  104401  005262                  2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
```

```
                                                TYPE    ,$QUES          ;TYPE (?) AND (CR),(LF)
                                                BR      ASKMDE          ;GO ASK AGAIN
015354 022737 000124 005262     4$:             CMP     #'T,BUFFO       ;SEE IF (T) TYPED
015362 001002                                   BNE     6$              ;BR IF NOT (T)
015364 000137 015672                            JMP     TYPLST          ;JUMP TO THE TYPE LIST ROUTINE
015370 022737 000117 005262     6$:             CMP     #'O,BUFFO       ;SEE IF (O) TYPED
015376 001002                                   BNE     8$              ;BR IF NOT (O)
015400 000137 016144                            JMP     OPNLST          ;JUMP TO THE OPEN LIST ROUTINE
015404 022737 000123 005262     8$:             CMP     #'S,BUFFO       ;SEE IF (S) TYPED
015412 001002                                   BNE     10$             ;BR IF NOT (S)
015414 000137 016430                            JMP     SETPRM          ;JUMP TO THE SET INDIV. PARAM. ROUTINE
015420 022737 000122 005262     10$:            CMP     #'R,BUFFO       ;SEE IF (R) TYPED
015426 001345                                   BNE     2$              ;BR IF NOT (R), TO ECHO BAD INPUT
015430 023737 005726 005730                     CMP     S0,S1           ;COMPARE 20(DEC) SECTOR LIMITS
015436 003403                                   BLE     12$             ;BR IF S0 NOT > S1
015440 104401 010651                            TYPE    ,SECNL1         ;TYPE "S0>S1 NOT ALLOWED"
015444 000717                                   BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
015446 023737 005732 005734     12$:            CMP     S2,S3           ;COMPARE 22(DEC) SECTOR LIMITS
015454 003405                                   BLE     14$             ;BR IF S2 NOT > S3
015456 104401 010701                            TYPE    ,SECNL2         ;TYPE "S2>S3"
015462 104401 010662                            TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
015466 000706                                   BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
015470 023737 005720 005722     14$:            CMP     FT,LT           ;COMPARE CHOSEN TRACK LIMITS
015476 003405                                   BLE     20$             ;BR IF FT NOT > LT
015500 104401 010713                            TYPE    ,TRKNLW         ;TYPE "FT>LT"
015504 104401 010662                            TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
015510 000675                                   BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
                                        ;CHECK WORD COUNT AGAINST PHYSICAL MEMORY AVAILABLE
015512 013700 005746            20$:            MOV     WC,R0           ;WORD COUNT LO BITS
015516 005300                                   DEC     R0              ;DECREMENT BY 1
015520 005001                                   CLR     R1              ;WORD COUNT HI BITS
015522 000241                                   CLC                     ;CONVERT WORD COUNT TO BYTES
015524 006100                                   ROL     R0
015526 006101                                   ROL     R1
015530 063700 005742                            ADD     MA,R0           ;ADD WORD COUNT (WC) TO MEM. ADR. (MA)
015534 005501                                   ADC     R1
015536 063701 005744                            ADD     MA+2,R1
015542 020137 006110                            CMP     R1,MAHILM+2     ;COMPARE HI BITS
015546 101004                                   BHI     26$             ;BR IF WC TOO BIG
015550 001006                                   BNE     28$             ;BR IF WC IS OK
015552 020037 006106                            CMP     R0,MAHILM       ;COMPARE LO BITS
015556 101404                                   BLOS    28$             ;BR IF WC IS OK
015560 104401 010725            26$:            TYPE    ,WC2BIG         ;TYPE "**  WC OR MA TOO LARGE"
015564 000647                                   BR      ASKMDE          ;GO BACK TO ASK AGAIN FOR PARAMS
                                        ;SAVE XXDP LOADER, IF NECESSARY
015566 004737 026556            28$:            JSR     PC,CHKXDP       ;GO SAVE XXDP LOADER IF NECESSARY
015572 015600                                   36$                     ;"NO SAVE" RETURN ADDRESS
015574 000137 016664            30$:            JMP     RUNTST          ;JUMP TO THE RUN TESTS ROUTINE
015600 105737 000041            36$:            TSTB    @#41            ;SEE IF LOADED BY XXDP
015604 001773                                   BEQ     30$             ;BR IF NOT
015606 104401 007333                            TYPE    ,OVLODR         ;TYPE "OVERLAY LOADER ? (Y OR N) *"
015612 004737 030174                            JSR     PC,RDCHRS       ;READ RESPONSE
015616 015600                                   36$                     ;(^C) RETURN ADDRESS
015620 015600                                   36$                     ;(^Z) RETURN ADDRESS
015622 015600                                   36$                     ;(^U) OR ERROR RETURN ADDRESS
015624 022700 000001                            CMP     #1,R0           ;SEE IF 1 CHAR TYPED
```

```
                        015630  001405                          BEQ     38$             ;BR IF 1 CHAR TYPED
                        015632  104401  005262          37$:    TYPE    .BUFFO          ;ECHO BAD INPUT
                        015636  104401  001314                  TYPE    ,$QUES
                        015642  000756                          BR      36$             ;GO ASK AGAIN
                        015644  122737  000131  005262  38$:    CMPB    #'Y,BUFFO       ;SEE IF (Y) TYPED
                        015652  001001                          BNE     40$             ;BR IF NOT (Y)
                        015654  000747                          BR      30$             ;GO RUN TESTS
                        015656  122737  000116  005262  40$:    CMPB    #'N,BUFFO       ;SEE IF (N) TYPED
                        015664  001362                          BNE     37$             ;BR IF NOT (N)
                        015666  000137  015304                  JMP     ASKMDE          ;GO ASK FOR PARAMETERS AGAIN


                                                        .SBTTL  TD - TYPE (T) LIST ROUTINE
                                                        ;*THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
                                                        ;*CONSOLE, LINE BY LINE. EACH LINE HAS THE FOLLOWING
                                                        ;*FORMAT:  XX=YYYYYY , WHERE XX IS A PARAMETER MNEMONIC
                                                        ;*AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
                                                        ;*TYPING (↑C) CAUSES IMMEDIATE RETURN TO DRVTST.
                                                        ;*AND (↑Z) CAUSES RETURN TO ASKMDE.

                        015672                          TYPLST:
                        015672  005001                          CLR     R1              ;INITIALIZE INDEX
                        015674  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                        015700  004737  030542          1$:     JSR     PC,TYPPRM       ;TYPE CURRENT PARAMETER AND VALUE
                        015704  104401  001315                  TYPE    ,$CRLF          ;TYPE (CR),(LF)
                        015710  005737  005522                  TST     INTCHR          ;SEE IF ANY INPUT AT KBD
                        015714  001420                          BEQ     8$              ;BR IF NO INPUT
                        015716  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
                        015724  001002                          BNE     4$              ;BR IF NOT (↑C)
                        015726  000137  013660                  JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
                        015732  122737  000032  005522  4$:     CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
                        015740  001002                          BNE     6$              ;BR IF NOT (↑Z)
                        015742  000137  015304                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
                        015746  004737  025350          6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
                        015752  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                        015756  022761  040515  006132  8$:     CMP     #"MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
                        015764  001002                          BNE     10$             ;BR IF NOT (MA)
                        015766  062701  000002                  ADD     #2,R1           ;INCREMENT PARAMETER INDEX
                        015772  062701  000002          10$:    ADD     #2,R1           ;INCREMENT PARAMETER INDEX
                        015776  010102                          MOV     R1,R2           ;GET COPY OF INDEX
                        016000  062702  005712                  ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
                        016004  022702  005756                  CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
                        016010  001333                          BNE     1$              ;BR IF NOT DONE YET
                        016012  032737  100000  005740          BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
                        016020  001005                          BNE     12$             ;BR IF PATTERN SPECIFIED
                        016022  042777  000100  163114          BIC     #BIT6,@#TKS     ;DISABLE KBD INTERRUPT
                        016030  000137  015304                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
                                                        ;TYPE OUT USER-DEFINED PATTERN 15
                        016034  104401  011040          12$:    TYPE    ,PFIFTN         ;TYPE "USER-DEFINED PATTERN 15 :"
                        016040  005001                          CLR     R1              ;INITIALIZE WORD INDEX
                        016042  005737  005522          14$:    TST     INTCHR          ;SEE IF ANY INPUT
                        016046  001420                          BEQ     20$             ;BR IF NO INPUT
                        016050  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
                        016056  001002                          BNE     16$             ;BR IF NOT (↑C)
                        016060  000137  013660                  JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
```

```
 4994  016064  122737  000032  005522  16$:    CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
 4995  016072  001002                          BNE     18$             ;BR IF NOT (↑Z)
 4996  016074  000137  015304                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 4997  016100  004737  025350          18$:    JSR     PC,ECOBAD       ;ECHO BAD INPUT
 4998  016104  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 4999  016110  004737  030622          20$:    JSR     PC,TYPPAT       ;TYPE WORD XX = YYYYYY
 5000  016114  104401  001315                  TYPE    ,$CR_F          ;TYPE (CR),(LF)
 5001  016120  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
 5002  016124  022701  000040                  CMP     #32.,R1         ;SEE IF 16 WORDS TYPED
 5003  016130  001344                          BNE     14$             ;BR IF NOT DONE YET
 5004  016132  042777  000100  163004          BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 5005  016140  000137  015304                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5006
 5007
 5008
 5009
 5010
 5011
 5012
 5013
 5014
 5015
 5016
 5017  016144                  .SBTTL  TD - OPEN (O) LIST ROUTINE
 5018  016144                  ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
 5019  016144                  ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
 5020  016144                  ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
 5021  016144                  ;*TTY INPUT. TYPING (↑C) CAUSES IMMEDIATE RETURN TO
 5022  016144                  ;*DRVTST, AND (↑Z) CALSES RETURN TO ASKMDE.
 5023  016144
 5024  016144                  OPNLST:
 5025  016144  104401  010756          TYPE    ,DFQUES         ;ASK IF ALL DEFAULT VALUES DESIRED
 5026  016150  004737  030174          JSR     PC,RDCHRS       ;READ RESPONSE TO DEFAULT QUESTION
 5027  016154  013660                  DRVTST                  ;(↑C) RETURN ADDRESS
 5028  016156  015304                  ASKMDE                  ;(↑Z) RETURN ADDRESS
 5029  016160  016144                  OPNLST                  ;(↑U) OR ERROR RETURN ADDRESS
 5030  016162  005700                  TST     R0              ;SEE IF NULL INPUT
 5031  016164  001433                  BEQ     NOTDFT          ;BR IF DEFAULTS NOT REQUESTED
 5032  016166  022737  000104  005262  CMP     #'D,BUFFO       ;SEE IF (D) TYPED
 5033  016174  001405                  BEQ     LODFPT          ;BR IF DEFAULTS DESIRED
 5034  016176  104401  005262          TYPE    ,BUFFO          ;ECHO BAD INPUT
 5035  016202  104401  001314          TYPE    ,$QUES
 5036  016206  000756                  BR      OPNLST          ;GO ASK AGAIN
 5037                  ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
 5038  016210  012700  005756  LODFPT: MOV     #PRDFLT,R0      ;LOAD DEFAULT LIST ADDRESS
 5039  016214  012702  005712          MOV     #PRMLST,R2      ;LOAD PARAMETER LIST ADDRESS
 5040  016220  012022          4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
 5041  016222  022702  005756          CMP     #PRDFLT,R2      ;SEE IF DONE YET
 5042  016226  001374                  BNE     4$              ;BR IF MORE VALUES TO LOAD YET
 5043  016230  105737  003106          TSTB    MDFLAG          ;SEE IF DEFAULT MODE
 5044  016234  001005                  BNE     6$              ;BR IF NOT DEFAULT MODE
 5045  016236  012737  000001  025016  MOV     #1,$EOPCT       ;SET PASS COUNT = 1
 5046  016244  000137  016746          JMP     STPASS          ;GO RUN DFLT TESTS
 5047  016250  000137  015304  6$:     JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5048  016254  005001          NOTDFT: CLR     R1              ;INITIALIZE INDEX
 5049                  ;TYPE CURRENT PARAMETER AND VALUE
 5050  016256  004737  030542  8$:     JSR     PC,TYPPRM       ;TYPE PARAMETER AND VALUE
 5051  016262  104401  012430          TYPE    ,SPACE1         ;TYPE A SPACE
 5052  016266  104401  012450          TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
 5053                  ;READ AND CHECK INPUT, IF ANY
 5054  016272  004737  030174          JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
 5055  016276  013660                  DRVTST                  ;(↑C) RETURN ADDRESS FOR RDCHRS
 5056  016300  015304                  ASKMDE                  ;(↑Z) RETURN ADDRESS FOR RDCHRS
```

```
016302  016256                          8$                      ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
016304  005700                          TST     R0              ;SEE IF ANY INPUT
016306  001007                          BNE     10$             ;BR IF ANY INPUT
016310  022761  040515  006132          CMP     #"MA,PRMNEM(R1) ;SEE IF (MA) JUST DEFAULTED
016316  001023                          BNE     12$             ;BR IF NOT (MA)
016320  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
016324  000420                          BR      12$             ;BR TO MOVE ON TO NEXT PARAMETER
016326  022700  000010          10$:    CMP     #10,R0          ;SEE IF 8 CHARACTERS TYPED
016332  002431                          BLT     14$             ;BR IF MORE THAN 8 TYPED
016334  012746  005262                  MOV     #BUFFO,-(SP)    ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
016340  004737  051702                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
016344  016416                          14$                     ;ERROR RETURN ADDRESS FOR OCTBIN
016346  012637  005466                  MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
016352  013737  052034  005470          MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
                                ;CHECK PARAMETER VALUE AND PUT INTO LIST
016360  004737  031200                  JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAM VALUE
016364  016416                          14$                     ;ERROR RETURN ADDR. FOR CHKPRM
                                ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
016366  004737  030654          12$:    JSR     PC,MODP15
                                ;MOVE ON TO NEXT PARAMETER
016372  062701  000002                  ADD     #2,R1           ;INCREMENT THE PARAMETER INDEX
016376  010102                          MOV     R1,R2           ;GET COPY OF INDEX
016400  062702  005712                  ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
016404  022702  005756                  CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
016410  001322                          BNE     8$              ;BR IF NOT DONE YET
016412  000137  015304                  JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
016416  104401  005262          14$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
016422  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
016426  000713                          BR      8$              ;BR TO ASK AGAIN



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

016430                          SETPRM:
016430  104401  012453                  TYPE    ,PRMPSP         ;TYPE "> " PROMPTER
                                ;READ AND CHECK INPUT, IF ANY
016434  004737  030174                  JSR     PC,RDCHRS       ;READ INPUT LINE
016440  013660                          DRVTST                  ;(↑C) RETURN ADDRESS FOR RDCHRS
016442  015304                          ASKMDE                  ;(↑Z) RETURN ADDRESS FOR RDCHRS
016444  016430                          SETPRM                  ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
016446  020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST 4 CHARACTERS TYPED
016452  002005                          BGE     6$              ;BR IF AT LEAST 4 TYPED
016454  104401  005262          4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
016460  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
016464  000761                          BR      SETPRM          ;BR TO ASK FOR INPUT AGAIN
016466  020027  000013          6$:     CMP     R0,#11.         ;SEE IF ELEVEN OR LESS CHARS TYPED
```

```
                                        BGT     4$                      ;BR IF MORE THAN ELEVEN TYPED
                                ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
                                        CLR     R1                      ;INITIALIZE PARAMETER INDEX
005262  006132          8$:     CMP     BUFFO,PRMNEM(R1)        ;TRY TO MATCH 2-CHAR MNEMONIC
                                        BEQ     10$                     ;BR IF PARAMETER FOUND IN TABLE
000002                          ADD     #2,R1                   ;INCREMENT INDEX
                                        MOV     R1,R2                   ;GET COPY OF INDEX
005712                          ADD     #PRMLST,R2              ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
005756                          CMP     #PRDF_T,R2
                                        BNE     8$                      ;BR IF MORE TO CHECK YET
                                        BR      4$                      ;BR TO ECHO BAD INPUT
                                ;CHECK FOR VALID LINE FORMAT
000002                  10$:    MOV     #2,R2                   ;INITIALIZE BUFFER POINTER
000075  005262                  CMPB    #'=,BUFFO(R2)           ;SEE IF NEXT CHAR IS "="
                                        BEQ     12$                     ;BR IF IT IS "="
000040  005262                  CMPB    #040,BUFFO(R2)          ;SEE IF NEXT CHAR IS A SPACE
                                        BNE     4$                      ;BR TO ECHO BAD INPUT
                                        INC     R2                      ;INCREMENT BUFFER POINTER
000075  005262                  CMPB    #'=,BUFFO(R2)           ;SEE IF NEXT CHAR IS "="
                                        BNE     4$                      ;BR TO ECHO INVALID INPUT
                                12$:    INC     R2                      ;INCREMENT BUFFER POINTER
                                        CMP     R2,R0                   ;SEE IF MORE CHARS LEFT IN BUFFER
                                        BGE     4$                      ;BR TO ECHO INVALID INPUT
000040  005262                  CMPB    #040,BUFFO(R2)          ;SEE IF NEXT CHARACTER IS SPACE
                                        BNE     14$                     ;BR IF NOT A SPACE
                                        INC     R2                      ;INCREMENT BUFFER POINTER
                                        CMP     R2,R0                   ;SEE IF MORE CHARS LEFT IN BUFFER
                                        BGE     4$                      ;BR IF NONE LEFT
                                14$:    SUB     R2,R0                   ;SUBTRACT POINTER FROM CHAR COUNT
000010                          CMP     R0,#10                  ;SEE HOW MANY CHARS LEFT
                                        BGT     4$                      ;BR IF TOO MANY LEFT
                                ;CHECK FOR LEGAL PARAMETER VALUE
005262                          ADD     #BUFFO,R2               ;GET ADDRESS OF DIGITS
                                        MOV     R2,-(SP)                ;PUT IT ON STACK FOR OCTBIN
051702                          JSR     PC,OCTBIN               ;CHECK DIGITS AND CONVERT TO BINARY
016454                          4$                              ;ERROR RETURN ADDR. FOR OCTBIN
005466                          MOV     (SP)+,LOWOCT            ;GET LOW BINARY BITS
052034  005470                  MOV     $HIOCT,HIGOCT           ;GET HIGH BINARY BITS
031200                          JSR     PC,CHKPRM               ;CHECK VALIDITY OF PARAMETER VALUE
016454                          4$                              ;ERROR RETURN ADDR. FOR CHKPRM
                                ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
030654                          JSR     PC,MODP15
000662                          BR      SETPRM                  ;RETURN TO ASK FOR ANOTHER PARAMETER



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

```
                                        ;*EACH SUCCESSIVE DRIVE FOR TESTING, UNTIL THE ENTIRE PASS
                                        ;* (ALL DRIVES) IS COMPLETED.
                                        
016664                                  RUNTST:
                                        ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
016664  104401  011207                4$:     TYPE    ,ENTPAS         ;ASK FOR NUMBER OF PASSES
016670  004737  030174                        JSR     PC,RDCHRS       ;READ RESPONSE
016674  013660                              DRVTST                  ;(↑C) RETURN ADDRESS
016676  015304                              ASKMDE                  ;(↑Z) RETURN ADDRESS
016700  016664                              4$                      ;(↑U) OR ERROR RETURN ADDRESS
016702  005700                              TST     R0              ;SEE IF NULL INPUT
016704  001403                              BEQ     6$              ;GO ASK AGAIN
016706  022700  000005                      CMP     #5,R0           ;SEE HOW MANY CHARS TYPED
016712  002005                              BGE     8$              ;BR IF 5 OR LESS
016714  104401  005262                6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
016720  104401  001314                        TYPE    ,$QUES
016724  000757                              BR      4$              ;GO ASK AGAIN
016726  012746  005262                8$:     MOV     #BUFFO,-(SP)    ;GET BUF ADDR ON STACK FOR OCTBIN
016732  004737  051702                        JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
016736  016714                              6$                      ;ERROR RETURN ADDRESS FOR OCTBIN
016740  012637  025016                      MOV     (SP)+,$EOPCT    ;SET DESIRED NO. OF PASSES (1-77777)
016744  001763                              BEQ     6$              ;BR IF 0, TO ASK AGAIN
                                        ;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
016746  005037  001326                STPASS: CLR     $PASS           ;INIT. THE PASS NUMBER
016752  112737  000001  003110              MOVB    #1,TSTING       ;SET "RUNNING TESTS" FLAG
016760  012737  177777  005500        NEWPAS: MOV     #177777,DRIVE   ;INITIALIZE DRIVE NO. TO -1
016766  005037  001330                      CLR     $DEVCT          ;INIT APT DEVICE COUNT TO 0
                                        ;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
016772  012737  000000  177776        NEWDRV: MOV     #PRO,@#PS       ;RE-ESTABLISH PRIORITY 0
017000  012706  001100                      MOV     #STACK,SP       ;RESTORE THE STACK
017004  105037  003120                      CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
017010  005037  005522                      CLR     INTCHR          ;CLEAR TTY INPUT BUFFER WORD
017014  005237  005500                      INC     DRIVE           ;INCREMENT DRIVE NUMBER
017020  022737  000010  005500            CMP     #10,DRIVE       ;SEE IF DONE WITH THIS PASS
017026  001002                            BNE     2$              ;BR IF NOT DONE CHECKING LIST
017030  000137  024764                    JMP     DUNPAS          ;JUMP IF DONE WITH THIS PASS
017034  013700  005500                2$:     MOV     DRIVE,R0        ;GET NEW DRIVE NUMBER
017040  105760  005576                      TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
017044  001752                            BEQ     NEWDRV          ;BR IF NOT MARKED, TO CHECK ANOTHER
017046  105037  001102                      CLRB    $TSTNM          ;INIT TEST NUMBER TO 0
017052  005037  001304                      CLR     $TIMES          ;INIT ITERATION COUNT
017056  010037  001332                      MOV     R0,$UNIT        ;SET DRIVE NO. FOR APT
                                        ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
017062  004737  027300                      JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
017066  112765  000125  000001            MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
017074  004737  037662                      JSR     PC,DRVCAL       ;DO READ HEADER
017100  132765  000020  000007            BITB    #B.CFMT,P.CS1H(R5) ;COMPLEMENT THE FORMAT BIT
017106  001404                            BEQ     4$              ;  GIVEN TO THE CONTROLLER,
017110  142765  000020  000007            BICB    #B.CFMT,P.CS1H(R5) ;  AND DO READ HEADER. THIS
017116  000403                            BR      6$              ;  WILL CAUSE SECTOR 0 HEADER
017120  152765  000020  000007    4$:     BISB    #B.CFMT,P.CS1H(R5) ;  TO BE READ.
017126  004737  037662                6$:     JSR     PC,DRVCAL       ;READ HEADER 0
017132  105037  003115                      CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
017136  013737  005732  005506            MOV     S2,FS           ;INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
017144  013737  005734  005510            MOV     S3,LS
017152  005762  000024                    TST     RKDB(R2)        ;POP SILO ONE TIME
```

```
5218  017156  032762  001000  000024          BIT     #BIT9,RKDB(R2)  ;TEST FORMAT BIT IN HEADER WORD 2
5219  017164  001411                          BEQ     8$              ;BR IF 22 SECTOR FORMAT
5220  017166  152737  000020  003115          BISB    #8.CFMT,FORMAT  ;SET 20 SECTOR FORMAT FOR THIS DRIVE
5221  017174  013737  005726  005506          MOV     S0,FS           ;INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
5222  017202  013737  005730  005510          MOV     S1,LS
5223  017210  013737  005752  005502  8$:     MOV     ST,STALLS       ;SET NUMBER OF UNIT STALLS DESIRED
5224  017216  004737  027300                  JSR     PC,INITSS       ;INIT THE S.S.
5225  017222  004737  036242                  JSR     PC,REDBSF       ;READ BAD SECTOR FILE FOR THIS DRIVE
5226  017226  113737  005500  011341          MOVB    DRIVE,DRVNO     ;GET DRIVE NO.
5227  017234  152737  000060  011341          BISB    #'0,DRVNO       ;CONVERT TO ASCII
5228  017242  104401  011320                  TYPE    ,TSTDRN         ;TYPE "TESTING DRIVE X"
5229  017246  005737  001326                  TST     $PASS           ;SEE IF THIS IS FIRST PASS
5230  017252  001012                          BNE     10$             ;BR IF NOT FIRST PASS
5231  017254  004737  031504                  JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
5232  017260  004737  031624                  JSR     PC,CRTSER       ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
5233  017264  032737  000020  005750          BIT     #BIT4,CS        ;SEE IF BAD SECTORS SHOULD BE TYPED
5234  017272  001402                          BEQ     10$             ;BR IF NOT
5235  017274  004737  036530                  JSR     PC,TYPBSF       ;TYPE BAD SECTOR FILES
5236  017300  005037  005522          10$:    CLR     INTCHR          ;INIT. TTY INPUT CHAR BUFFER
5237  017304  105737  003126                  TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
5238  017310  001402                          BEQ     TST1            ;BR IF NOT
5239  017312  000137  024040                  JMP     RWDTST          ;JUMP TO R/W DATA TEST
5240
5241
5242
```

```
                                        ;;*****************************************************************
                                        ;*TEST 1        RECALIBRATE/SEEK TEST
                                        ;*
                                        ;*      THIS TEST PERFORMS A RECALIBRATE, FOLLOWED BY A
                                        ;*      SEEK TO CYLINDER LC.
                                        ;*
                                        ;*****************************************************************
017316  000004                    TST1:   SCOPE
017320  012737  000001  001324            MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
017326  004737  030032                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
017332  004737  030100                    JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
017336  000137  017376                    JMP     TST2            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
017342  004737  027300                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
017346  004737  025330                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;RECALIBRATE THE DRIVE
017352  112765  000113  000001            MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
017360  004737  037662                    JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
                                        ;DO SEEK TO CYLINDER LC
017364  013765  005714  000002            MOV     LC,P.CYLN(R5)   ;SET CYLINDER = LC
017372  004737  031742                    JSR     PC,SEEKER       ;DO A READ HEADER OR EXPLICIT SEEK



                                        ;;*****************************************************************
                                        ;*TEST 2        SEEK/SEEK TEST
                                        ;*
                                        ;*      THIS TEST PERFORMS A SEEK TO CYLINDER FC, FOLLOWED
                                        ;*      BY A SEEK TO CYLINDER LC.
                                        ;*
                                        ;*****************************************************************
017376  000004                    TST2:   SCOPE
017400  012737  000002  001324            MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
017406  004737  030032                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
017412  004737  030100                    JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
017416  000137  017456                    JMP     TST3            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
017422  004737  027300                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
017426  004737  025330                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;DO SEEK TO CYLINDER FC
017432  013765  005712  000002            MOV     FC,P.CYLN(R5)   ;SET CYL = FC
017440  004737  031742                    JSR     PC,SEEKER       ;SEEK TO FC
                                        ;DO SEEK TO CYLINDER LC
017444  013765  005714  000002            MOV     LC,P.CYLN(R5)   ;SET CYL = LC
017452  004737  031742                    JSR     PC,SEEKER       ;SEEK TO LC



                                        ;;*****************************************************************
                                        ;*TEST 3        CYLINDER ADDRESSING TEST
                                        ;*
                                        ;*      THIS TEST VERIFIES THE CYLINDER ADDRESS BITS, BY
                                        ;*      SEEKING TO CYLINDERS 0,1,2,4,10,20,40,100,200,400(OCT).
                                        ;*      THEN, THE SEEKS ARE DONE IN REVERSE, TO 400,200,100,
                                        ;*      40,20,10,4,2,1,0.
                                        ;*
```

```
                                        ;;*************************************************************
017456  000004                          TST3:   SCOPE
017460  012737  000003  001324                  MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
017466  004737  030032                          JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
017472  004737  030100                          JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
017476  000137  017564                          JMP     TST4            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
017502  004737  027300                          JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
017506  004737  025330                          JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;SEEK TO CYLINDER 0
017512  005065  000002                          CLR     P.CYLN(R5)      ;SET CYL = 0
017516  004737  031742                          JSR     PC,SEEKER       ;SEEK TO CYL 0
                                        ;SEEK TO CYLINDERS 1,2,4,10,20,40,100,200,400 (FORWARD DIRECTION)
017522  005265  000002                          INC     P.CYLN(R5)      ;INITIALIZE CYL TO 1
017526  004737  031742                  2$:     JSR     PC,SEEKER       ;SEEK TO CURRENT CYL
017532  006365  000002                          ASL     P.CYLN(R5)      ;GET NEXT CYL NO.
017536  032765  001000  000002                  BIT     #BIT09,P.CYLN(R5) ;SEE IF DONE WITH FORWARD SEEKS
017544  001770                                  BEQ     2$              ;BR IF NOT DONE YET
                                        ;SEEK TO CYLINDERS 400,200,100,40,20,10,4,2,1,0 (REVERSE DIRECTION)
017546  006265  000002                  4$:     ASR     P.CYLN(R5)      ;GET NEXT CYL NO.
017552  004737  031742                          JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
017556  005765  000002                          TST     P.CYLN(R5)      ;SEE IF DONE IN REVERSE DIRECTION
017562  001371                                  BNE     4$              ;BR IF NOT DONE YET
```

```
                                        ;;*************************************************************
                                        ;*TEST 4        CYLINDER ADDRESS CROSSTALK TEST
                                        ;*
                                        ;*      THIS TEST PERFORMS SEEKS TO CYLINDERS WHOSE ADDRESSES
                                        ;*      ARE SUSCEPTIBLE TO BIT CROSSTALK WITHIN THE CYLINDER
                                        ;*      ADDRESSING LOGIC. THE CYLINDER ADDRESS BITS SEQUENCE
                                        ;*      AS FOLLOWS :
                                        ;*              000 000 000
                                        ;*              011 111 111
                                        ;*              000 000 000
                                        ;*              011 111 110
                                        ;*              000 000 000
                                        ;*              011 111 101
                                        ;*              000 000 000
                                        ;*                   .
                                        ;*                   .
                                        ;*                   .
                                        ;*              000 000 000
                                        ;*              010 111 111
                                        ;*              000 000 000
                                        ;*              101 111 111
                                        ;*
                                        ;*************************************************************
017564  000004                          TST4:   SCOPE
017566  012737  000004  001324                  MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
017574  004737  030032                          JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
017600  004737  030100                          JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
017604  000137  017730                          JMP     TST5            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
017610  004737  027300                          JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
```

```
5355  017614  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5356                                  ;SEEK TO CYL 0
5357  017620  005065  000002                  CLR     P.CYLN(R5)      ;SET CYL = 0
5358  017624  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYLINDER 0
5359                                  ;SEEK TO CYL 377 (OCT)
5360  017630  012765  000377  000002          MOV     #377,P.CYLN(R5) ;SET CYL = 377
5361  017636  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYL 377
5362                                  ;SEEK TO REMAINING CROSSTALK CYLINDERS
5363  017642  012737  000001  005504          MOV     #1,CYLNDR       ;INITIALIZE BIT MASK
5364  017650  005065  000002          2$:     CLR     P.CYLN(R5)      ;SET CYL = 0
5365  017654  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYL 0
5366  017660  032737  000200  005504          BIT     #BIT07,CYLNDR   ;SEE IF ABOUT TO DO LAST SEEK
5367  017666  001013                          BNE     4$              ;BR IF THIS WILL BE LAST SEEK
5368  017670  012765  000377  000002          MOV     #377,P.CYLN(R5) ;SET CYL = 377
5369  017676  043765  005504  000002          BIC     CYLNDR,P.CYLN(R5) ;ZERO A BIT TO PROMOTE CROSSTALK
5370  017704  004737  031742                  JSR     PC,SEEKER       ;SEEK TO THIS CYLINDER
5371  017710  006337  005504                  ASL     CYLNDR          ;SHIFT BIT MASK
5372  017714  000755                          BR      2$              ;GO SEEK TO CYL 0 AGAIN
5373  017716  012765  000577  000002  4$:     MOV     #577,P.CYLN(R5) ;SET CYL = 577
5374  017724  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYL 577
5375
5376
5377
5378                                  ;;*****************************************************************
5379                                  ;*TEST 5        INCREMENT/DECREMENT SEEK TEST
5380                                  ;*
5381                                  ;*      IN THIS TEST SEEKS ARE DONE IN INCREMENTS OF IC CYLINDERS
5382                                  ;*      STARTING AT CYL FC, AND ENDING AT OR BEYOND LC. THEN, THE
5383                                  ;*      SEEKS ARE DONE IN REVERSE, BACK TO FC. IF FC IS CHOSEN > LC,
5384                                  ;*      THE SEEKS WILL PROCEED IN THE OPPOSITE DIRECTION.
5385                                  ;*
5386                                  ;;*****************************************************************
5387  017730  000004                  TST5:   SCOPE
5388  017732  012737  000005  001324          MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5389  017740  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5390  017744  004737  030100                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5391  017750  000137  020134                  JMP     TST6            ;JUMP TO NEXT TEST
5392                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5393  017754  004737  027300                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5394  017760  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5395                                  ;SEEK IN INCREMENTS, FROM FC TO LC
5396  017764  013737  005712  005512          MOV     FC,NCYL1        ;INITIALIZE NCYL1 TO FC
5397  017772  013765  005512  000002  2$:     MOV     NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5398  020000  004737  031742                  JSR     PC,SEEKER       ;SEEK TO NCYL1
5399  020004  023737  005712  005714          CMP     FC,LC           ;SEE IF FC > LC
5400  020012  003010                          BGT     4$              ;BR IF FC > LC
5401  020014  063737  005716  005512          ADD     IC,NCYL1        ;INCREMENT NCYL1
5402  020022  023737  005714  005512          CMP     LC,NCYL1        ;SEE IF LC EXCEEDED
5403  020030  002360                          BGE     2$              ;BR IF NOT YET
5404  020032  000407                          BR      6$              ;LC REACHED - DO REVERSE SEEKS
5405  020034  163737  005716  005512  4$:     SUB     IC,NCYL1        ;DECREMENT NCYL1
5406  020042  023737  005714  005512          CMP     LC,NCYL1        ;SEE IF LC EXCEEDED
5407  020050  003750                          BLE     2$              ;BR IF NOT YET
5408                                  ;SEEK IN INCREMENTS, BACK TO FC
5409  020052  023737  005712  005714  6$:     CMP     FC,LC           ;SEE IF FC >LC
5410  020060  003010                          BGT     8$              ;BR IF FC > LC
```

```
5411  020062  163737  005716  005512          SUB     IC,NCYL1          ;DECREMENT NCYL1
5412  020070  023737  005712  005512          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5413  020076  003410                          BLE     10$               ;BR IF NOT YET
5414  020100  000415                          BR      12$               ;FC REACHED - ALL DONE
5415  020102  063737  005716  005512  9$:     ADD     IC,NCYL1          ;INCREMENT NCYL1
5416  020110  023737  005712  005512          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5417  020116  002406                          BLT     12$               ;FC REACHED - ALL DONE
5418  020120  013765  005512  000002  10$:     MOV     NCYL1,P.CYLN(R5)  ;SET CYL = NCYL1
5419  020126  004737  031742                  JSR     PC,SEEKER         ;SEEK TO NCYL1
5420  020132  000747                          BR      6$                ;CONTINUE
5421  020134                          12$:
5422
5423
5424                                  ;;*****************************************************************
5425                                  ;*TEST 6        OSCILLATING SEEK TEST
5426                                  ;*
5427                                  ;*      IN THIS TEST, SEEKS OF INCREASING LENGTH ARE DONE FROM FC
5428                                  ;*      TO NCYL1, AND BACK TO FC. NCYL1 IS INCREMENTED BY IC AFTER
5429                                  ;*      EACH SEEK, AND THE SEEKS TO NCYL1 ALWAYS PROCEED FROM FC
5430                                  ;*      TOWARD LC (FC > LC IS PERMISSIBLE). WHEN LC IS REACHED OR
5431                                  ;*      EXCEEDED, THE SAME SEEKS ARE DONE IN REVERSE ORDER, UNTIL
5432                                  ;*      NCYL1 EXCEEDS FC.
5433                                  ;*
5434                                  ;*****************************************************************
5435  020134  000004                  TST6:   SCOPE
5436  020136  012737  000006  001324          MOV     #6,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
5437  020144  004737  030032                  JSR     PC,SETTUP         ;SET UP FOR LOOP ON ERROR
5438  020150  004737  030100                  JSR     PC,CHKITR         ;SEE IF ITER. NO. = 0 FOR THIS TEST
5439  020154  000137  020364                  JMP     TST7              ;JUMP TO NEXT TEST
5440                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5441  020160  004737  027300                  JSR     PC,INITSS         ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5442  020164  004737  025330                  JSR     PC,PREPKB         ;PREPARE FOR POSSIBLE KBD INPUT
5443                          ;DO OSCILLATING SEEKS TOWARD LC
5444  020170  013737  005712  005512          MOV     FC,NCYL1          ;INITIALIZE NCYL1 TO FC
5445  020176  013765  005512  000002  2$:     MOV     NCYL1,P.CYLN(R5)  ;SET CYL = NCYL1
5446  020204  004737  031742                  JSR     PC,SEEKER         ;SEEK TO NCYL1
5447  020210  013765  005712  000002          MOV     FC,P.CYLN(R5)     ;SET CYL = FC
5448  020216  004737  031742                  JSR     PC,SEEKER         ;SEEK TO FC
5449  020222  023737  005712  005714          CMP     FC,LC             ;SEE IF FC > LC
5450  020230  003010                          BGT     4$                ;BR IF FC > LC
5451  020232  063737  005716  005512          ADD     IC,NCYL1          ;INCREMENT NCYL1
5452  020240  023737  005714  005512          CMP     LC,NCYL1          ;SEE IF LC EXCEEDED
5453  020246  002353                          BGE     2$                ;BR IF NOT YET
5454  020250  000407                          BR      6$                ;LC REACHED - DO REVERSE SEEKS
5455  020252  163737  005716  005512  4$:     SUB     IC,NCYL1          ;DECREMENT NCYL1
5456  020260  023737  005714  005512          CMP     LC,NCYL1          ;SEE IF LC EXCEEDED
5457  020266  003743                          BLE     2$                ;BR IF NOT YET
5458                          ;DO OSCILLATING SEEKS TOWARD FC
5459  020270  023737  005712  005714  6$:     CMP     FC,LC             ;SEE IF FC >LC
5460  020276  003010                          BGT     8$                ;BR IF FC > LC
5461  020300  163737  005716  005512          SUB     IC,NCYL1          ;DECREMENT NCYL1
5462  020306  023737  005712  005512          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5463  020314  003410                          BLE     10$               ;BR IF NOT YET
5464  020316  000422                          BR      12$               ;FC REACHED - ALL DONE
5465  020320  063737  005716  005512  8$:     ADD     IC,NCYL1          ;INCREMENT NCYL1
5466  020326  023737  005712  005512          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
```

```
020334  002413                          BLT     12$                     ;FC REACHED - ALL DONE
020336  013765  005512  000002  10$:    MOV     NCYL1,P.CYLN(R5)        ;SET CYL = NCYL1
020344  004737  031742                  JSR     PC,SEEKER               ;SEEK TO NCYL1
020350  013765  005712  000002          MOV     FC,P.CYLN(R5)           ;SET CYL = FC
020356  004737  031742                  JSR     PC,SEEKER               ;SEEK TO FC
020362  000742                          BR      6$                      ;CONTINUE
020364                          12$:

                                ;;*****************************************************************
                                ;*TEST 7         CONVERGING/DIVERGING SEEK TEST
                                ;*
                                ;*       THIS TEST PERFORMS SEEKS WHICH EXERCISE CYLINDER DIFFERENCE
                                ;*       VALUES. INITIALLY, NCYL1 = FC AND NCYL2 = LC. A SEEK IS
                                ;*       DONE TO NCYL1 AND THEN TO NCYL2. THEN, NCYL1 IS INCREMENTED
                                ;*       BY IC IN THE DIRECTION OF LC, AND NCYL2 IS INCREMENTED BY
                                ;*       IC IN THE DIRECTION OF FC, AND SEEKS ARE DONE TO NCYL1
                                ;*       AND NCYL2 AGAIN. THIS SEEKING IS CONTINUED WITH INCREMENTED
                                ;*       VALUES OF NCYL1 AND NCYL2, UNTIL NCYL1 EXCEEDS LC AND NCYL2
                                ;*       EXCEEDS FC. (NOTE : FC > LC IS PERMISSIBLE.)
                                ;*       THIS TEST CAUSES SEEKING TO CONVERGING AND THEN DIVERGING
                                ;*       CYLINDER VALUES.
                                ;*
                                ;*****************************************************************
020364  000004                  TST7:   SCOPE
020366  012737  000007  001324          MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
020374  004737  030032                  JSR     PC,SETTUP               ;SET UP FOR LOOP ON ERROR
020400  004737  030100                  JSR     PC,CHKITR               ;SEE IF ITER. NO. = 0 FOR THIS TEST
020404  000137  020542                  JMP     TST10                   ;JUMP TO NEXT TEST
                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
020410  004737  027300                  JSR     PC,INITSS               ;INITIALIZE DRIVER PARAMS AND SUB-SYS
020414  004737  025330                  JSR     PC,PREPKB               ;PREPARE FOR POSSIBLE KBD INPUT
020420  013737  005712  005512          MOV     FC,NCYL1                ;SET NCYL1 = FC
020426  013737  005714  005514          MOV     LC,NCYL2                ;SET NCYL2 = LC
020434  013765  005512  000002  2$:     MOV     NCYL1,P.CYLN(R5)        ;SET CYL = NCYL1
020442  004737  031742                  JSR     PC,SEEKER               ;SEEK TO NCYL1
020446  013765  005514  000002          MOV     NCYL2,P.CYLN(R5)        ;SET CYL = NCYL2
020454  004737  031742                  JSR     PC,SEEKER               ;SEEK TO NCYL2
020460  023737  005712  005714          CMP     FC,LC                   ;SEE IF FC > LC
020466  003013                          BGT     4$                      ;BR IF FC > LC
020470  063737  005716  005512          ADD     IC,NCYL1                ;INCREMENT NCYL1
020476  163737  005716  005514          SUB     IC,NCYL2                ;DECREMENT NCYL2
020504  023737  005714  005512          CMP     LC,NCYL1                ;SEE IF LIMITS EXCEEDED
020512  002350                          BGE     2$                      ;BR IF NOT YET
020514  000412                          BR      6$                      ;LIMITS REACHED - ALL DONE
020516  163737  005716  005512  4$:     SUB     IC,NCYL1                ;DECREMENT NCYL1
020524  063737  005716  005514          ADD     IC,NCYL2                ;INCREMENT NCYL2
020532  023737  005714  005512          CMP     LC,NCYL1                ;SEE IF LIMITS EXCEEDED
020540  003735                          BLE     2$                      ;BR IF NOT YET
020542                          6$:

                                ;;*****************************************************************
                                ;*TEST 10        PSEUDO-RANDOM SEEK TEST
                                ;*
```

```
                                          ;*      THIS TEST PERFORMS A SEEK TO A PSEUDO-RANDOMLY CHOSEN
                                          ;*      CYLINDER, WHICH IS WITHIN THE RANGE (0-632).
                                          ;*
                                          ;;*************************************************************
 020542  000004                           TST10:  SCOPE
 020544  012737  000010  001324                   MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 020552  004737  030032                           JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
 020556  004737  030100                           JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
 020562  000137  020614                           JMP     TST11           ;JUMP TO NEXT TEST
                                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
 020566  004737  027300                           JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
 020572  004737  025330                           JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 020576  004737  032122                           JSR     PC,RNDADR       ;SELECT A PSEUDO-RANDOM CYLINDER
 020602  013765  005504  000002                   MOV     CYLNDR,P.CYLN(R5)  ;SET CYL VALUE
 020610  004737  031742                           JSR     PC,SEEKER       ;SEEK TO THIS CYLINDER



                                          ;;*************************************************************
                                          ;*TEST 11        MAXIMUM VELOCITY REVERSAL SEEK TEST
                                          ;*
                                          ;*      THIS TEST PERFORMS A SEEK TO CYLINDER 0, FOLLOWED BY A
                                          ;*      SEEK TO CYL 201 (OCT), AND THEN A RETURN SEEK TO 0. THIS
                                          ;*      OPERATION REQUIRES THE HEADS TO DECELERATE UPON REACHING
                                          ;*      THE POINT OF MAXIMUM VELOCITY, AND INDUCES HEATING IN
                                          ;*      THE SERVO MECHANISM.
                                          ;*
                                          ;;*************************************************************
 020614  000004                           TST11:  SCOPE
 020616  012737  000011  001324                   MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 020624  004737  030032                           JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
 020630  004737  030100                           JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
 020634  000137  020702                           JMP     TST12           ;JUMP TO NEXT TEST
                                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
 020640  004737  027300                           JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
 020644  004737  025330                           JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 020650  005065  000002                           CLR     P.CYLN(R5)      ;SET CYL = 0
 020654  004737  031742                           JSR     PC,SEEKER       ;SEEK TO CYL 0
 020660  012765  000201  000002                   MOV     #201,P.CYLN(R5) ;SET CYL = 201 (OCT)
 020666  004737  031742                           JSR     PC,SEEKER       ;SEEK TO CYL 201
 020672  005065  000002                           CLR     P.CYLN(R5)      ;SET CYL = 0
 020676  004737  031742                           JSR     PC,SEEKER       ;SEEK TO CYL 0



                                          ;;*************************************************************
                                          ;*TEST 12        MECHANICAL VIBRATION SEEK TEST
                                          ;*
                                          ;*      THIS TEST PERFORMS GEOMETRICALLY INCREASING SEEKS WITH
                                          ;*      GEOMETRICALLY DECREASING STALL TIME BETWEEN OPERATIONS
                                          ;*      WITH INTENT TO INDUCE VARYING VIBRATIONAL MODES UPON
                                          ;*      THE DRIVE. THE TEST BEGINS WITH LC = 1, AND ST = SM. THEN,
                                          ;*      THE FOLLOWING SEQUENCE IS PERFORMED :
                                          ;*      SEEKS ARE DONE BETWEEN 0 AND LC, TEN TIMES. THEN, ST IS
                                          ;*      DIVIDED BY 2 AND LC IS DOUBLED, AND SEEKS ARE DONE BETWEEN
                                          ;*      0 AND LC AGAIN, TEN TIMES.THIS PROCESS IS CONTINUED FOR NEW
```

```
                                        T12     MECHANICAL VIBRATION SEEK TEST

                                        ;*      VALUES OF ST AND LC, UNTIL LC EXCEEDS CYL 400 (OCT). THEN,
                                        ;*      THE WHOLE PROCESS IS REVERSED, WITH ST BEING DOUBLED AND
                                        ;*      LC DIVIDED BY 2, UNTIL LC BECOMES < 1.
                                        ;*
                                        ;;**********************************************************************
005584  020702  000004                  ST12:   SCOPE
005585  020704  012737  000012  001324          MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
005586  020712  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
005587  020716  004737  030100                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
005588  020722  000137  021142                  JMP     TST13           ;JUMP TO NEXT TEST
005589                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
005590  020726  004737  027300                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
005591  020732  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
005592  020736  012737  000001  005504          MOV     #1,CYLNDR       ;INIT. CURRENT CYL TO 1
005593  020744  013737  005754  001276          MOV     SM,$TMP6        ;INIT. STALLS TO SM
005594                                  ;DO INCREASING SEEKS
005595  020752  012737  000012  005532  2$:     MOV     #10.,SCRACH     ;INIT. LOOP COUNTER TO 10(DEC)
005596  020760  013737  001276  005502          MOV     $TMP6,STALLS    ;SET CURRENT STALL IN STALL WORD
005597  020766  005065  000002          4$:     CLR     P.CYLN(R5)      ;SET CYL = 0
005598  020772  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYL 0
005599  020776  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET CURRENT CYLINDER
005600  021004  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
005601  021010  005337  005532                  DEC     SCRACH          ;DECREMENT LOOP COUNTER
005602  021014  001364                          BNE     4$              ;BR IF NOT 5 TIMES YET
005603  021016  000241                          CLC                     ;CLEAR GARBAGE BIT
005604  021020  006037  001276                  ROR     $TMP6           ;DIVIDE STALLS BY 2
005605  021024  006037  001300                  ROR     $TMP7           ;SAVE THE DROPPED STALL BIT
005606  021030  006337  005504                  ASL     CYLNDR          ;DOUBLE THE CURRENT CYLINDER NUMBER
005607  021034  032737  001000  005504          BIT     #BIT09,CYLNDR   ;SEE IF DONE WITH INCREASING SEEKS YET
005608  021042  001743                          BEQ     2$              ;BR IF NOT DONE YET
005609                                  ;DO DECREASING SEEKS
005610  021044  006237  005504          6$:     ASR     CYLNDR          ;DIVIDE CURRENT CYLINDER BY 2
005611  021050  006137  001300                  ROL     $TMP7           ;GET A SAVED STALL BIT INTO CARRY
005612  021054  006137  001276                  ROL     $TMP6           ;DOUBLE THE STALL
005613  021060  012737  000012  005532          MOV     #10.,SCRACH     ;INIT. LOOP COUNTER TO 10(DEC)
005614  021066  013737  001276  005502          MOV     $TMP6,STALLS    ;SET CURRENT STALL IN STALL WORD
005615  021074  005065  000002          8$:     CLR     P.CYLN(R5)      ;SET CYL = 0
005616  021100  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CYL 0
005617  021104  013765  005504  000002          MOV     CYLNDR,P.CYLN(R5) ;SET CURRENT CYL NUMBER
005618  021112  004737  031742                  JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
005619  021116  005337  005532                  DEC     SCRACH          ;DECREMENT LOOP COUNTER
005620  021122  001364                          BNE     9$              ;BR IF NOT 5 TIMES YET
005621  021124  032737  000002  005504          BIT     #BIT01,CYLNDR   ;SEE IF DONE YET
005622  021132  001744                          BEQ     6$              ;BR IF NOT DONE YET
005623  021134  013737  005752  005502          MOV     ST,STALLS       ;RESTORE DESIRED NO. OF UNIT STALLS
005624
005625
005626
005627
005628
005629
005630
005631
005632
005633
005634  021142  000004                  ;;**********************************************************************
                                        ;*TEST 13        MAX ROTATIONAL LATENCY MEASUREMENT
                                        ;*THIS TEST MEASURES THE INTERVAL OF TIME BETWEEN 2 INDEX MARKS
                                        ;*IN A SINGLE DISK ROTATION. THE SPECIFIED MAXIMUM ROTATIONAL
                                        ;*LATENCY = 25 MILLI-SEC + OR - 2.5%. THIS MEASUREMENT IS MADE
                                        ;* 128 TIMES.
                                        ;;**********************************************************************
                                        ST13:   SCOPE
```

```
                                        ;SET TEST NUMBER IN APT MAIL BOX
                                        ;SET UP FOR LOOP ON ERROR
                                        ;SEE IF ITER. NO. = 0 FOR THIS TEST
                                        ;JUMP TO NEXT TEST
                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
                                        ;INITIALIZE DRIVER PARAMS AND SUB-SYS
                                        ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;SEE IF TIMING TESTS ARE ALLOWED
                                        ;BR IF TEST ALLOWED
                                        ;SKIP TO NEXT TEST
                        26$:            ;CALIBRATE SOFTWARE TIMER
                        1$:
                                        ;ERROR RETURN ADDR FOR CALBRT
                                        ;INIT VARIABLES TO BE USED
                                        ;INHIBIT ↑C, ↑Z ESCAPE
                                        ;INIT MEASUREMENT COUNTER
                        ;READ HEADER TO INITIALIZE FORMAT BIT IN CONTROLLER
                                        ;SET FORMAT BIT = 0
                                        ;SET READ HDR COMMAND
                                        ;DO READ HDR TO SET FORMAT TO 0
                                        ;ALLOW ↑C, ↑Z ESCAPE
                        ;CHANGE FORMAT AND READ HDR TO FIND INDEX
                        2$:             ;INIT RK06 INTR INDICATOR
                                        ;SET SPECIAL RK06 HANDLER ADDRESS
                                        ;INIT TIME-OUT INDICATOR
                                        ;COMPLEMENT FORMAT BIT TO 1
                                        ;COMPLEMENT FORMAT BIT TO 1
                                        ;GET PARAMETER BLK ADDR
                                        ;STORE PREV AND CURRENT COMMANDS
                                        ;INITIATE A READ HDR TO FIND INDEX
                        4$:             ;STORE PARAM BLK ADDRESS HERE
                        6$:             ;SEE IF RK06 INTR REC'D YET
                                        ;BR IF RK06 INTR REC'D
                                        ;INCREMENT TIME-OUT INDICATOR
                                        ;BR IF NO TIME-OUT
                        7$:             ;GET COMMAND FOR REPORT
                                        ;GATHER STATUS
                                        ;TIMED-OUT ON READ HEADER
                                        ;GO TO EXIT
                        28$:            ;RESTORE R0-R5
                                        ;TAKE ERROR EXIT
                        ;CHANGE FORMAT AND READ HDR TO MEASURE TIME TO NEXT INDEX
                        3$:             ;INIT RK06 INTERRUPT INDICATOR
                                        ;INIT HI CYCLE COUNT
                                        ;SAVE PARAM BLK ADDRESS
                                        ;COMPLEMENT FORMAT BIT TO 0
                                        ;COMPLEMENT FORMAT BIT TO 0
                                        ;ISSUE READ HDR TO FIND NEXT INDEX
                                        ;MEASURE THE TIME TO NEXT INDEX
                                        ;ERROR RETURN ADDRESS
                        ;CONVERT MEAS'D CYCLES TO TIME AND CHECK AGAINST LIMITS
                                        ;SAVE R0-R5
                                        ;CONVERT MEAS'D CYCLES TO TIME
                                        ;ERROR RETURN ADDRESS
```

```
021426  060337  003166                  ADD     R3,SUMLO1       ;ADD CONVERTED TIME TO SUM
021432  005537  003170                  ADC     SUMHI1
021436  060237  003170                  ADD     R2,SUMHI1
021442  004737  035700                  JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
                                ;COMPARE R2-R3 AGAINST CURRENT MIN AND MAX MEASUREMENTS
021446  020237  003204                  CMP     R2,MINHI1       ;COMP HI BITS TO HI MIN
021452  101010                          BHI     32$             ;BR IF > MIN
021454  103403                          BLO     30$             ;BR IF < MIN
021456  020337  003202                  CMP     R3,MINLI1       ;COMP LO BITS TO LO MIN
021462  103004                          BHIS    32$             ;BR IF > OR = MIN
021464  010237  003204          30$:    MOV     R2,MINHI1       ;SET NEW MIN
021470  010337  003202                  MOV     R3,MINLI1
021474  020237  003210          32$:    CMP     R2,MAXIHI       ;COMP HI BITS TO HI MAX
021500  103410                          BLO     14$             ;BR IF < MAX
021502  101003                          BHI     34$             ;BR IF > MAX
021504  020337  003206                  CMP     R3,MAXIL1       ;COMP LO BITS TO LO MAX
021510  101404                          BLOS    14$             ;BR IF < OR = MAX
021512  010237  003210          34$:    MOV     R2,MAXIH1       ;SET NEW MAX
021516  010337  003206                  MOV     R3,MAXIL1
                                ;COMPARE MEAS'D TIME TO SPECIFIED LIMITS
021522  020227  000000          14$:    CMP     R2,#MINHI1      ;COMPARE HI BITS TO HI MINIMUM
021526  101006                          BHI     18$             ;BR IF > MIN
021530  103403                          BLO     16$             ;BR IF < MIN
021532  020327  057467                  CMP     R3,#MINLO1      ;COMPARE LO BITS TO LO MIN
021536  103002                          BHIS    18$             ;BR IF > OR = MIN
021540  005237  003160          16$:    INC     BLWMIN          ;INCREMENT COUNT OF TIMES BELOW MIN
021544  020227  000000          18$:    CMP     R2,#MAXHI1      ;COMPARE HI BITS TO HI MAX
021550  103406                          BLO     22$             ;BR IF < MAX
021552  101003                          BHI     20$             ;BR IF > MAX
021554  020327  062031                  CMP     R3,#MAXLO1      ;COMPARE LO BITS TO LO MAX
021560  101402                          BLOS    22$             ;BR IF < OR = MAX
021562  005237  003162          20$:    INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE MAX
021566  104410                  22$:    RESREG                  ;RESTORE R0-R5
021570  005203                          INC     R3              ;INCREMENT MEASUREMENT COUNTER
021572  013705  005532                  MOV     SCRACH,R5       ;RESTORE PARAM BLK ADDRESS
021576  022703  000200                  CMP     #128.,R3        ;SEE IF 128 MEASUREMENTS YET
021602  001225                          BNE     2$              ;BR IF NOT DONE WITH MEASUREMENTS YET
                                ;DVIDE SUM BY 128. TO GET AVERAGE TIME OF ROTATION
021604  012704  000007                  MOV     #7,R4           ;SET LOOP COUNTER = 7
021610  000241                  24$:    CLC                     ;DIVIDE DOUBLE-WORD SUM BY 2
021612  006037  003170                  ROR     SUMHI1
021616  006037  003166                  ROR     SUMLO1
021622  005304                          DEC     R4              ;SEE IF DONE WITH DIVISION
021624  001371                          BNE     24$             ;BR IF NOT DONE YET
                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
021626  032737  000010  005750          BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
021634  001036                          BNE     36$             ;BR IF REPORTS SHOULD BE INHIBITED
021636  104401  011552                  TYPE    ,ROTIMS         ;TYPE "ROTATIONAL TIMES :"
021642  012701  003202                  MOV     #MINIL1,R1      ;POINTER TO LO MIN
021646  004737  033556                  JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
021652  013746  003160                  MOV     BLWMIN,-(SP)    ;GET COUNT OF TIMES BELOW MIN
021656  104405                          TYPDS                   ;CONVERT AND TYPE IT
021660  104401  012025                  TYPE    ,BELOW          ;TYPE " OF 128 BELOW MIN OF "
021664  104401  012270                  TYPE    ,LIM1           ;TYPE SPEC'D MIN LIMIT
021670  012701  003206                  MOV     #MAXIL1,R1      ;POINTER TO LO MAX
021674  004737  033602                  JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
```

```
021700  013746  003162                MOV     ABVMX1,-(SP     ;GET COUNT OF TIMES ABOVE MAX
021704  104405                      TYPDS                   ;CONVERT AND TYPE IT
021706  104401  012062              TYPE    .ABOVE          ;TYPE " OF 128 ABOVE MAX OF "
021712  104401  012303              TYPE    ,LIM2           ;TYPE SPEC'D MAX LIMIT
021716  012701  003166              MOV     #SUML01,R1      ;POINTER TO AVG TIME
021722  004737  033626              JSR     PC,TYPAVG       ;TYPE AVERAGE TIME
021726  104401  001315              TYPE    ,$CRLF          ;TYPE <CR>,<LF>
021732  012777  044536  16:070  36$: MOV     #I.INTR,@RKVEC  ;RESTORE RK06 HANDLER ADDRESS
021740  000177  177250              JMP     @1$             ;PROCEED TO NEXT TEST

                        ;;*****************************************************************
                        ;*SPECIAL INTERRUPT HANDLER FOR READ HEADERS IN ROT. LAT. MEAS. TEST
                        ;;*****************************************************************
021744  005200          RHEDHD: INC     R0              ;SET RK06 INTR HANDLER
021746  005762  000000          TST     RKCS1(R2)       ;SEE IF CONTROLLER ERROR SET
021752  100002                  BPL     2$              ;BR IF NO ERROR
021754  000137  044536          JMP     I.INTR          ;LET DRIVER PROCESS THE ERROR
021760  000002          2$:     RTI                     ;RETURN FROM INTR

                        ;;*****************************************************************
                        ;*TEST 14       ONE CYLINDER SEEK TIME MEASUREMENT
                        ;*THIS TEST MEASURES THE TIME REQUIRED TO SEEK BETWEEN 2 AJACENT
                        ;*CYLINDERS, BOTH IN THE FORWARD AND REVERSE DIRECTIONS. SEEKS
                        ;*ARE DONE TO 0,1,2,...,631,632  AND THEN TO 631,630,...,2,1,0
                        ;* AND THE RESULTS ARE TYPED FOR EACH DIRECTION.
                        ;*THE SPECIFIED ONE CYL SEEK TIME IS < 8 MILLI-SEC.
                        ;;*****************************************************************
021762  000004          TST14:  SCOPE
021764  012737  000014  001324          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021772  004737  030032              JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021776  004737  030100              JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
022002  000137  022302              JMP     TST15           ;JUMP TO NEXT TEST
                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
022006  004737  027300              JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
022012  004737  025330              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
022016  105737  003123              TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
022022  001002                  BNE     2$              ;BR IF ALLOWED
022024  000177  000004          1$:     JMP     @4$             ;SKIP TO NEXT TEST
022030  004737  032260          2$:     JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
022034                          4$:
022034  022302                          TST15                   ;ERROR RETURN ADDR FOR CALBRT
022036  004737  032552              JSR     PC,INIVRB       ;INIT VARIABLES USED
022042  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
022050  005065  000002              CLR     P.CYLN(R5)      ;INIT CYL TO 0
022054  004737  037662              JSR     PC,DRVCAL       ;DO INITIAL SEEK TO CYL 0
                        ;MEASURE FORWARD SEEK TIME
022060  005265  000002          6$:     INC     P.CYLN(R5)      ;INCREMENT CYLINDER BY 1
022064  004737  032660              JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
022070  022302                          TST15                   ;ERROR RETURN ADDR FOR CALBRT
022072  060137  003166              ADD     R1,SUML01       ;ADD MEAS'D TIME TO FORWARD SUM
022076  005537  003170              ADC     SUMHI1
022102  060037  003170              ADD     R0,SUMHI1
022106  004737  035700              JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
                        ;COMPARE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
```

```
5803  022112  012703  003202              MOV     #MINIL1,R3      ;SET POINTER TO CURRENT MIN AND MAX
5804  022116  004737  033172              JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5805                              ;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
5806  022122  020027  000000              CMP     R0,#MAXHI2      ;COMPARE HI BITS TO HI MAX
5807  022126  103406                      BLO     12$             ;BR IF < MAX
5808  022130  101003                      BHI     10$             ;BR IF > MAX
5809  022132  020127  017500              CMP     R1,#MAXLO2      ;COMPARE LO BITS TO LO MAX
5810  022136  101402                      BLOS    12$             ;BR IF < OR = MAX
5811  022140  005237  003162      10$:    INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
5812  022144  022765  000632  000002  12$:    CMP     #632,P.CYLN(R5) ;SEE IF LAST FORWARD SEEK DONE
5813  022152  001342                      BNE     6$              ;BR IF NOT DONE
5814                              ;MEASURE A REVERSE SEEK TIME
5815  022154  005365  000002      14$:    DEC     P.CYLN(R5)      ;DECREMENT CYL BY 1
5816  022160  004737  032660              JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
5817  022164  022302                      TST15                   ;ERROR RETURN ADDR FOR CALBRT
5818  022166  060137  003172              ADD     R1,SUMLO2       ;ADD MEASURED TIME TO REVERSE SUM
5819  022172  005537  003174              ADC     SUMHI2
5820  022176  060037  003174              ADD     R0,SUMHI2
5821  022202  004737  035700              JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
5822                              ;COMPARE REVERSE TIME TO CURRENT REVERSE MIN AND MAX
5823  022206  012703  003212              MOV     #MINIL2,R3      ;SET POINTER TO CURRENT MIN AND MAX
5824  022212  004737  033172              JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5825                              ;COMPARE REVERSE TIME TO SPEC'D MAX LIMIT
5826  022216  020027  000000              CMP     R0,#MAXHI2      ;COMPARE HI BITS TO HI MAX
5827  022222  103406                      BLO     18$             ;BR IF < MAX
5828  022224  101003                      BHI     16$             ;BR IF > MAX
5829  022226  020127  017500              CMP     R1,#MAXLO2      ;COMPARE LO BITS TO LO MAX
5830  022232  101402                      BLOS    18$             ;BR IF < OR = MAX
5831  022234  005237  003164      16$:    INC     ABVMX2          ;INCR COUNT OF TIMES ABOVE SPEC'D MAX
5832  022240  005765  000002      18$:    TST     P.CYLN(R5)      ;SEE IF LAST REVERSE SEEK DONE
5833  022244  001343                      BNE     14$             ;BR IF NOT DONE YET
5834                              ;COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES
5835  022246  012703  000632              MOV     #632,R3         ;GET NO. OF MEASUREMENTS
5836  022252  004737  033244              JSR     PC,GETAVG       ;COMPUTE AVERAGES
5837                              ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
5838  022256  032737  000010  005750      BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5839  022264  001257                      BNE     1$              ;BR IF REPORTS SHOULD BE INHIBITED
5840  022266  104401  011632              TYPE    ,ONECYL         ;TYPE "ONE CYL SEEK TIMES"
5841  022272  012703  012117              MOV     #ABOVE1,R3      ;GET POINTER TO MAX SPEC'D LIMIT MSG
5842  022276  004737  033336              JSR     PC,TYPTMS       ;TYPE TIMING TEST RESULTS
5843
5844
5845
5846                              ;;*****************************************************************
5847                              ;*TEST 15       AVERAGE SEEK TIME MEASUREMENT
5848                              ;*THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD
5849                              ;*AND REVERSE DIRECTIONS. THE AVERAGE TIME IS CALCULATED FROM THE
5850                              ;*FOLLOWING FORMULA :
5851                              ;*  T AVG = [T1(410)(2)+T2(409)(2)+...+T410(1)(2)]/(410)(410)
5852                              ;*  WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS.
5853                              ;*FORWARD AND REVERSE TIMES ARE MEASURED AND TYPED,
5854                              ;*SEPARATELY.
5855                              ;*THE SPECIFIED AVERAGE SEEK TIME IS < 38 MILLI-SEC.
5856                              ;;*****************************************************************
5857  022302  000004              TST15:  SCOPE
5858  022304  012737  000015  001324      MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
                                                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
                                                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
                                                JMP     TST16           ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
                                                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
                                                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                                TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
                                                BNE     2$              ;BR IF ALLOWED
                                        1$:     JMP     4$              ;SKIP TO NEXT TEST
                                        2$:     JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
                                        4$:
                                                TST16                   ;ERROR RETURN ADDR FOR CALBRT
                                                JSR     PC,INIVRB       ;INIT VARIABLES
                                                MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
                                                CLR     P.CYLN(R5)      ;SET CYL = 0
                                                JSR     PC,DRVCAL       ;SEEK INITIALLY TO 0
                                                CLR     NCYL1           ;SET DESTINATION CYLINDER
                                                MOV     #411.,SCRACH    ;INIT COEFFICIENT TO 411(DEC)
                                        8$:     INC     NCYL1           ;INCR DESTINATION CYL
                                                DEC     SCRACH          ;DECREMENT COEFFICIENT
                                                MOV     NCYL1,P.CYLN(R5)
                                                JSR     PC,TIMSEK       ;MEASURE A FORWARD SEEK TIME
                                                TST16                   ;ERROR RETURN ADDR FOR CALBRT
                                                JSR     PC,FDTERM       ;COMPUTE AND ADD A FORWARD TERM
                                                CLR     P.CYLN(R5)      ;SET CYL TO 0
                                                JSR     PC,TIMSEK       ;MEASURE A REVERSE SEEK TIME
                                                TST16                   ;ERROR RETURN ADDR FOR CALBRT
                                                JSR     PC,RVTERM       ;COMPUTE AND ADD A REVERSE TERM
                                                CMP     NCYL1,#410.     ;SEE IF DONE MEASURING YET
                                                BLT     8$              ;BR IF NOT DONE YET
                                                TYPE    ,AVGSEK         ;TYPE "AVERAGE SEEK TIMES"

                                        ;DIVIDE FORWARD SUM BY (410)(410)
                                                MOV     SUML01,R0       ;GET DIVIDEND
                                                MOV     SUMHI1,R1
                                                MOV     SUML02,R2
                                                MOV     SUMHI2,R3
                                                MOV     #2,R4           ;GET DIVISOR
                                                MOV     #37028.,R5
                                                JSR     PC,M.DPID       ;PERFORM DIVISION
                                                MOV     R2,SUML01       ;GET FORWARD AVG
                                                MOV     R3,SUMHI1

                                        ;DIVIDE REVERSE SUM BY (410)(410)
                                                MOV     MAXIL1,R0       ;GET DIVIDEND
                                                MOV     MAXIH1,R1
                                                MOV     MAXIL2,R2
                                                MOV     MAXIH2,R3
                                                JSR     PC,M.DPID       ;PERFORM DIVISION
                                                MOV     R2,SUML02       ;GET REVERSE AVG
                                                MOV     R3,SUMHI2

                                        ;TYPE FORWARD AVERAGE
                                                TYPE    ,FORWRD         ;TYPE "***FORWARD DIRECTION***"
                                                MOV     #SUMHI1,R1      ;POINTER TO FORWARD AVG
                                                JSR     PC,TYPAVG       ;TYPE FORWARD AVERAGE
                                                TYPE    ,SPCDMX         ;TYPE "SPEC'D MAX IS 40000 US"

                                        ;TYPE REVERSE AVERAGE
                                                TYPE    ,REVRSE         ;TYPE "***REVERSE DIRECTION***"
```

```
5915  022620  012701  003174                MOV     #SUMHI2,R1      ;POINTER TO REVERSE AVG
5916  022624  004737  033626                JSR     PC,TYPAVG       ;TYPE REVERSE AVERAGE
5917  022630  104401  012234                TYPE    ,SPCDMX         ;TYPE "SPEC'D MAX IS 40000 US"
5918  022634  004737  027300                JSR     PC,INITSS       ;INIT THE SUB-SYS
5919
5920
5921                                  ;;*****************************************************************
5922                                  ;*TEST 16       MAXIMUM SEEK TIME MEASUREMENT
5923                                  ;*THIS TEST MEASURES THE TIME  REQUIRED TO SEEK FORM CYL 0 TO
5924                                  ;*CYL 632 OCT. (410 DEC.). THIS TIME REPRESENTS THE MAXIMUM SEEK
5925                                  ;*TIME, AND IT IS MEASURED 128 TIMES IN EACH DIRECTION.  THE
5926                                  ;*THE SPECIFIED MAXIMUM SEEK TIME IS < 75 MILLI-SEC.
5927                                  ;;*****************************************************************
5928  022640  000004                TST16:  SCOPE
5929  022642  012737  000016  001324        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5930  022650  004737  030032                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5931  022654  004737  030100                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5932  022660  000137  023164                JMP     TST17           ;JUMP TO NEXT TEST
5933                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5934  022664  004737  027300                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5935  022670  004737  025330                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5936  022674  105737  003123                TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
5937  022700  001002                        BNE     2$              ;BR IF ALLOWED
5938  022702  000177  000004          1$:   JMP     @4$             ;SKIP TO NEXT TEST
5939  022706  004737  032260          2$:   JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
5940  022712                          4$:
5941  022712  023164                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
5942  022714  004737  032552                JSR     PC,INIVRB       ;INIT VARIABLES USED
5943  022720  112765  000117  000001        MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5944  022726  005065  000002                CLR     P.CYLN(R5)      ;INIT CYL TO 0
5945  022732  004737  037662                JSR     PC,DRVCAL       ;DO INITIAL SEEK TO CYL 0
5946  022736  005037  005532                CLR     SCRACH          ;INIT MEASUREMENT COUNT
5947                                ;MEASURE FORWARD SEEK TIME
5948  022742  012765  000632  000002  6$:   MOV     #632,P.CYLN(R5) ;SET CYL = 632 OCT.
5949  022750  004737  032660                JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
5950  022754  023164                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
5951  022756  060137  003166                ADD     R1,SUML01       ;ADD MEAS'D TIME TO FORWARD SUM
5952  022762  005537  003170                ADC     SUMHI1
5953  022766  060037  003170                ADD     R0,SUMHI1
5954  022772  004737  035700                JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
5955                                ;COMPARE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
5956  022776  012703  003202                MOV     #MINIL1,R3      ;SET POINTER TO CURRENT MIN AND MAX
5957  023002  004737  033172                JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5958                                ;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
5959  023006  020027  000001                CMP     R0,#MAXHI4      ;COMPARE HI BITS TO HI MAX
5960  023012  103406                        BLO     12$             ;BR IF < MAX
5961  023014  101003                        BHI     10$             ;BR IF > MAX
5962  023016  020127  022370                CMP     R1,#MAXLO4      ;COMPARE LO BITS TO LO MAX
5963  023022  101402                        BLOS    12$             ;BR IF < OR = MAX
5964  023024  005237  003162          10$:  INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
5965                                ;MEASURE A REVERSE SEEK TIME
5966  023030  005065  000002          12$:  CLR     P.CYLN(R5)      ;SET CYL = 0
5967  023034  004737  032660                JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
5968  023040  023164                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
5969  023042  060137  003172                ADD     R1,SUML02       ;ADD MEASURED TIME TO REVERSE SUM
5970  023046  005537  003174                ADC     SUMHI2
```

```
5971  023052  060037  003174                ADD     R0,SUMHI2
5972  023056  004737  035700                JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
5973                                ;COMPARE REVERSE TIME TO CURRENT REVERSE MIN AND MAX
5974  023062  012703  003212                MOV     #MINIL2,R3      ;SET POINTER TO CURRENT MIN AND MAX
5975  023066  004737  033172                JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5976                                ;COMPARE REVERSE TIME TO SPEC'D MAX LIMIT
5977  023072  020027  000001                CMP     R0,#MAXHI4      ;COMPARE HI BITS TO HI MAX
5978  023076  103406                        BLO     18$             ;BR IF < MAX
5979  023100  101003                        BHI     16$             ;BR IF > MAX
5980  023102  020127  022370                CMP     R1,#MAXLC4      ;COMPARE LO BITS TO LO MAX
5981  023106  101402                        BLOS    18$             ;BR IF < OR = MAX
5982  023110  005237  003164        16$:    INC     ABVMX2          ;INCR COUNT OF TIMES ABOVE SPEC'D MAX
5983  023114  005237  005532        18$:    INC     SCRACH          ;INCREMENT MEASUREMENT COUNT
5984  023120  022737  000200  005532        CMP     #128.,SCRACH    ;SEE IF DONE WITH 128 MEASUREMENTS
5985  023126  001305                        BNE     6$              ;BR IF NOT DONE YET
5986                                ;COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES
5987  023130  012703  000200                MOV     #128.,R3        ;GET NO. OF MEASUREMENTS
5988  023134  004737  033244                JSR     PC,GETAVG       ;COMPUTE AVERAGES
5989                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
5990  023140  032737  000010  005750        BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5991  023146  001255                        BNE     1$              ;BR IF REPORTS SHOULD BE INHIBITED
5992  023150  104401  011663                TYPE    ,MAXSEK         ;TYPE "MAXIMUM SEEK TIMES"
5993  023154  012703  012165                MOV     #ABOVE3,R3      ;GET POINTER TO MAX SPEC'D LIMIT MSG
5994  023160  004737  033336                JSR     PC,TYPTMS       ;TYPE TIMING TEST RESULTS
5995
5996
5997
5998                                ;;*****************************************************************
5999                                ;*TEST 17       SECTOR ADDRESSING TEST
6000                                ;*IN THIS TEST, ALL SECTORS OF FC, FT ARE WRITTEN, EACH WITH
6001                                ;*256(DEC) WORDS OF ITS OWN SECTOR NO. + 100(OCT). NEXT, FOR
6002                                ;*EACH SECTOR WRITTEN A WRITE CHECK IS DONE, FOLLOWED BY A
6003                                ;*READ AND SOFTWARE COMPARE OF THE DATA.
6004                                ;;*****************************************************************
6005  023164  000004                TST17:  SCOPE
6006  023166  012737  000017  001324        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6007  023174  004737  030032                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6008  023200  004737  030100                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
6009  023204  000137  023604                JMP     TST20           ;JUMP TO NEXT TEST
6010                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
6011  023210  004737  027300                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
6012  023214  004737  025330                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
6013  023220  112765  000123  000001        MOVB    #WRDATA,P.CMND(R5)      ;SET WRITE DATA COMMAND
6014  023226  013765  005712  000002        MOV     FC,P.CYLN(R5)           ;SET CYL = FC
6015  023234  113765  005720  000005        MOVB    FT,P.TRCK(R5)           ;SET TRACK = FT
6016  023242  013737  005712  001174        MOV     FC,$REG5                ;SET PACK ADR FOR ERROR PRINTOUT
6017  023250  013737  005720  001176        MOV     FT,$REG6
6018  023256  005037  001200                CLR     $REG7
6019  023262  105065  000004                CLRB    P.SECT(R5)              ;SET SECTOR = 0
6020  023266  012765  063526  000010        MOV     #RWBUF,P.BALO(R5)       ;SET DATA BUFFER ADDRESS
6021  023274  012765  177400  000012        MOV     #-256.,P.WC(R5)         ;SET WORD CNT FOR 256(DEC) WORDS
6022                                ;SET NUMBER OF SECTORS IN R3
6023  023302  012703  000024                MOV     #20.,R3         ;SET R3 INITIALLY = 20(DEC)
6024  023306  105737  003115                TSTB    FORMAT          ;DETERMINE THE FORMAT
6025  023312  001002                        BNE     4$              ;BR IF 20 SECTOR FORMAT
6026  023314  012703  000026                MOV     #22.,R3         ;SET R3 = 22(DEC)
```

```
6027                                        ;WRITE THE CURRENT SECTOR WITH THE SECTOR NUMBER + 100(OCT)
6028  023320  116500  000004          4$:   MOVB    P.SECT(R5),R0   ;PUT SECTOR NO. INTO R0
6029  023324  062700  000100                ADD     #100,R0         ;ADD 100(OCT) TO SECTOR NO.
6030  023330  004737  033652                JSR     PC,LODSEC       ;LOAD THE ENTIRE BUFFER WITH SECTOR NO. + 100(OCT)
6031  023334  004737  037662                JSR     PC,DRVCAL       ;WRITE THE SECTOR
6032  023340  105265  000004                INCB    P.SECT(R5)      ;INCREMENT THE SECTOR NO.
6033  023344  120365  000004                CMPB    R3,P.SECT(R5)   ;SEE IF LAST SECTOR WRITTEN YET
6034  023350  001363                        BNE     4$              ;BR IF NOT DONE YET
6035  023352  012737  023352  001110        MOV     #..$LPERR       ;SET NEW LOOP ON ERROR ADRS
6036  023360  004737  030032                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6037  023364  105065  000004                CLRB    P.SECT(R5)      ;INITIALIZE SECTOR TO 0
6038                                        ;DO WRITE CHECK OF A SECTOR
6039  023370  116500  000004          8$:   MOVB    P.SECT(R5),R0   ;GET SECTOR INTO R0
6040  023374  062700  000100                ADD     #100,R0         ;ADD 100(OCT) TO SECTOR NO.
6041  023400  004737  033652                JSR     PC,LODSEC       ;LOAD BUFFER WITH SECTOR + 100(OCT.
6042  023404  112765  000131  000001        MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6043  023412  004737  037662                JSR     PC,DRVCAL       ;DO A WRITE CHECK
6044                                        ;READ THE SECTOR AND COMPARE THE DATA TO SECTOR NUMBER + 100(OCT)
6045  023416  112765  000121  000001        MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
6046  023424  004737  037662                JSR     PC,DRVCAL       ;READ THIS SECTOR INTO MEMORY
6047  023430  012701  063526                MOV     #RWBUF,R1       ;GET ADDR. OF DATA BUF INTO R1
6048  023434  005004                        CLR     R4              ;CLEAR WORD NUMBER
6049  023436  005037  005532                CLR     SCRACH          ;INIT. COMPARE ERROR COUNT
6050  023442  116537  000004  001200        MOVB    P.SECT(R5),$REG7
6051  023450  032737  000002  005474        BIT     #BSERR,RECODE   ;SEE IF BAD SECTOR ERROR OCCURRED
6052  023456  001045                        BNE     22$             ;BR IF YES
6053  023460  020021                10$:    CMP     R0,(R1)+        ;COMPARE BUF WORD TO SECTOR + 100(OCT)
6054  023462  001437                        BEQ     20$             ;BR IF NO COMPARE ERROR
6055  023464  005737  005532                TST     SCRACH          ;CHECK ERROR COUNT
6056  023470  001006                        BNE     12$             ;BR IF NOT FIRST ERROR IN SECTOR
6057  023472  105037  062045                CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
6058  023476  012737  000005  063420        MOV     #5,DF25+2       ;ADJ. ERROR DATA WORD COUNT
6059  023504  104034                        ERROR   34              ;TYPE HEADING FOR ERROR MSG
6060  023506  005237  005532          12$:  INC     SCRACH          ;INCREMENT ERROR COUNT
6061  023512  032777  000001  155420        BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
6062  023520  001004                        BNE     14$             ;BR TO REPORT ALL ERRORS
6063  023522  022737  000012  005532        CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
6064  023530  002420                        BLT     22$             ;BR IF > 10.
6065  023532  010437  001202          14$:  MOV     R4,$REG10       ;WORD NUMBER
6066  023536  010037  001204                MOV     R0,$REG11       ;GOOD DATA
6067  023542  016137  177776  001206        MOV     -2(R1),$REG12   ;BAD DATA
6068  023550  104063                        ERROR   63              ;TYPE GOOD AND BAD DATA
6069  023552  032777  000100  155360        BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
6070  023560  001004                        BNE     22$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
6071  023562  005204                20$:    INC     R4              ;INCREMENT WORD NUMBER
6072  023564  022704  000400                CMP     #256.,R4        ;SEE IF DONE YET
6073  023570  001333                        BNE     10$             ;BR IF NOT DONE COMPARING YET
6074  023572  105265  000004          22$:  INCB    P.SECT(R5)      ;INCREMENT SECTOR NO.
6075  023576  120365  000004                CMPB    R3,P.SECT(R5)   ;SEE IF ALL SECTORS CHECKED YET
6076  023602  001272                        BNE     8$              ;BR IF NOT DONE YET
6077
6078
6079
6080                                        ;;*****************************************************************
6081                                        ;*TEST 20       TRACK ADDRESSING TEST
6082                                        ;*IN THIS TEST, SECTOR F5 OF CYL FC IS WRITTEN WITH 256 (DEC) WORDS
```

```
                                        ;*OF THE TRACK NO. + 100(OCT), FOR EACH OF TRACKS 0,1,2. THEN, A
                                        ;*WRITE CHECK OF EACH SECTOR IS DONE TO VERIFY THE WRITES. THEN, EACH
                                        ;*OF THE 3 SECTORS IS RE-WRITTEN, AND AFTER EACH WRITE ALL OF THE
                                        ;*THREE SECTORS ARE WRITE-CHECKED TO DETECT TRACK ADDRESSING PROBLEMS.
                                        ;;*****************************************************************
023604  000004                  TST20:  SCOPE
023606  012737  000020  001324          MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
023614  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
023620  004737  030100                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
023624  000137  024040                  JMP     TST21           ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
023630  004737  027300                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
023634  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
023640  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
023646  013765  005712  000002          MOV     FC,P.CYLN(R5)   ;SET CYL = FC
023654  105065  000005                  CLRB    P.TRCK(R5)      ;INITIALIZE TRACK NO. TO 0
023660  113765  005506  000004          MOVB    FS,P.SECT(R5)   ;SET SECTOR =FS
023666  012765  063526  000010          MOV     #RWBUF,P.BALO(R5) ;SET DATA BUFFER ADDRESS
023674  012765  177400  000012          MOV     #-256.,P.WC(R5) ;SET WORD COUNT FOR 256(DEC) WORDS
                                        ;WRITE THE TRACK NO. + 100(OCT) AT SECTOR FS OF TRACKS 0,1,2
023702  116500  000005          4$:     MOVB    P.TRCK(R5),R0   ;SET TRACK NO. IN R0
023706  062700  000100                  ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
023712  004737  033652                  JSR     PC,LODSEC       ;LOAD BUFFER WITH TRACK NO. + 100(OCT)
023716  004737  037662                  JSR     PC,DRVCAL       ;WRITE THE SECTOR
023722  105265  000005                  INCB    P.TRCK(R5)      ;INCREMENT THE TRACK NO.
023726  122765  000003  000005          CMPB    #3,P.TRCK(R5)   ;SEE IF ALL 3 TRACKS WRITTEN YET
023734  001362                          BNE     4$              ;BR IF NOT DONE YET
                                        ;DO A WRITE-CHECK OF THE 3 SECTORS WRITTEN, TO VERIFY THE DATA
023736  012737  023736  001110          MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
023744  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
023750  004737  033674                  JSR     PC,TRKCHK
                                        ;WRITE A SECTOR AND WRITE-CHECK ALL 3 SECTORS
023754  012737  023754  001110          MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
023762  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
023766  105065  000005                  CLRB    P.TRCK(R5)      ;INIT TRACK TO 0
023772  116500  000005          6$:     MOVB    P.TRCK(R5),R0   ;GET TRACK NO. IN R0
023776  062700  000100                  ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
024002  004737  033652                  JSR     PC,LODSEC       ;LOAD BUF WITH TRACK + 100(OCT)
024006  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
024014  004737  037662                  JSR     PC,DRVCAL       ;RE-WRITE A SECTOR
024020  004737  033674                  JSR     PC,TRKCHK       ;WRITE-CHECK ALL 3 SECTORS
024024  105265  000005                  INCB    P.TRCK(R5)      ;INCR THE TRACK NO.
024030  122765  000003  000005          CMPB    #3,P.TRCK(R5)   ;SEE IF ALL 3 TRACKS RE-WRITTEN YET
024036  001355                          BNE     6$              ;BR IF NOT DONE YET


024040                          RWDTST:
        000040                  RWTINX=<$TN-1>*2                ;R/W TEST INDEX
                                        ;;*************************************************************
                                        ;*TEST 21       READ/WRITE DATA TEST
                                        ;*THE READ/WRITE DATA TEST HAS 2 DIFFERENT VERSIONS, DEPENDING
                                        ;*ON THE VALUE OF THE DATA PATTERN PARAMETER (PT).
                                        ;*
                                        ;*  FOR PT = 0 :
                                        ;*THIS TEST IS THE QUICK VERIFY DEFAULT DATA TEST, WHICH IS RUN
```

```
                                ;*WHEN PT=0, DUE TO PARAMETER INPUT CHOICE, OR ADDRESS 200 START.
                                ;*IN THIS TEST, THE ENTIRE PACK IS TESTED WITH A SINGLE DATA PATTERN.
                                ;*THIS PATTERN IS COMPRISED OF THE DATA PATTERNS 00-15, WHICH REPEATS
                                ;*EVERY 256(DEC) WORDS. EACH TRACK IS WRITTEN IN 4 SEGMENTS, FOLLOWED
                                ;*BY A WRITE-CHECK, READ, AND SOFTWARE COMPARE. THE SEGMENTS ARE 6
                                ;*SECTORS EACH, WHICH MEANS THAT SPIRALING OCCURRS ON THE LAST
                                ;*SEGMENT WRITTEN ON EACH TRACK.
                                ;*
                                ;*  FOR PT NOT = 0 :
                                ;*THIS TEST PERFORMS READ/WRITE FUNCTIONS ON THE ENTIRE RANGE OF
                                ;*CYLINDERS (FC-LC), TRACKS (FT-LT), AND SECTORS (FS-LS) SPECIFIED.
                                ;*AT EACH SPECIFIED SECTOR WC WORDS OF THE CURRENT REPEATING DATA
                                ;*PATTERN ARE WRITTEN, AND THEN WRITE-CHECKED, FOLLOWED BY A READ
                                ;*AND SOFTWARE COMPARE. THIS IS DONE FOR ALL THE SPEC'D SECTORS ON ALL
                                ;*THE TRACKS, USING SECTOR INCR IS, AND TRACK INCR IT.
                                ;*     THEN IT IS REPEATED USING EACH OF THE OTHER DATA
                                ;*PATTERNS CHOSEN IN PARAMETER PT. THEN, EACH OF THE ABOVE OPERATIONS
                                ;*ARE REPEATED AT EACH OF THE REMAINING CYLINDERS, IN THE SPECIFIED
                                ;*RANGE, USING THE CYLINDER INCREMENT IC.
                                ;* NOTE THAT FS MUST BE CHOSEN < OR = LS, AND FT MUST BE < OR = LT.
                                ;*HOWEVER, FC MAY BE <, =, OR > LC, AND IF FC>LC, THE CYLINDER
                                ;*ADDRESS WILL BE DECREMENTED BY IC (INSTEAD OF INCREMENTED) TO
                                ;*OBTAIN EACH NEW CYLINDER ADDRESS.
                                ;;*****************************************************************
024040  000004                  T$T21:  SCOPE
024042  012737  000021  001324          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
024050  004737  030032                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
024054  004737  030100                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
024060  000137  024752                  JMP     $EOP            ;JUMP TO END-OF-PASS ROUTINE
                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
024064  004737  027300                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
024070  004737  025330                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
024074  105037  003124                  CLRB    XOVLAD          ;INIT XXDP OVERLAID INDICATOR
024100  032777  000040  155032          BIT     #BITS,@SWR      ;SEE IF WRITES INHIBITED
024106  001002                          BNE     2$              ;IF YES,DON'T CHANGE RAND DATA PAT
024110  004737  033760                  JSR     PC,LODP14       ;GENERATE PSEUDO-RAND PAT 14
024114  013737  005740  005534  2$:     MOV     PT,PATRN        ;GET A COPY OF PT
024122  005737  005740                  TST     PT              ;SEE IF QUICK VERIFY DATA TEST DESIRED
024126  001020                          BNE     7$              ;BR IF NOT QUICK VERIFY
                                ;SET PARAMETERS FOR QUICK VERIFY DEFAULT DATA TEST
024130  012765  063526  000010          MOV     #RWBUF,P.BALO(R5) ;SET R/W BUFFER ADDRESS
024136  005065  000002                  CLR     P.CYLN(R5)      ;INIT CYL TO 0
024142  105065  000005          4$:     CLRB    P.TRCK(R5)      ;INIT TRACK TO 0
024146  012765  175000  000012  5$:     MOV     #-1536.,P.WC(R5) ;SET WORD COUNT FOR 6 SECTORS
024154  105065  000004                  CLRB    P.SECT(R5)      ;INIT SECTOR TO 0
024160  004737  035330          6$:     JSR     PC,CHKLIM       ;CHK WRD COUNT AND PACK ADR TO AVOID OVERFLOW
024164  024726                          30$                     ;RETURN ADDR FOR TRANSFER NOT ALLOWED
024166  000457                          BR      18$             ;PROCEED
                                ;SET PARAMETERS FOR NON-DEFAULT DATA TEST
024170  013765  005742  000010  7$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
024176  013700  005744                  MOV     MA+2,R0         ;GET MA BITS 16-21
024202  042700  177774                  BIC     #177774,R0      ;MASK FOR LO 2 BITS
024206  150065  000007                  BISB    R0,P.BAHI(R5)   ;SET BA BITS 16,17
024212  013737  005742  005572          MOV     MA,PMA          ;SET BUFFER ADDRESS
024220  013737  005744  005574          MOV     MA+2,PMA+2
024226  105237  003124                  INCB    XOVLAD          ;SET XXDP OVERLAID INDICATOR
```

```
024232  005737  055134                  TST     $KT11                   ;SEE IF MEMORY MANAGEMENT PRESENT
024236  100002                          BPL     9$                      ;BR IF NO MEM. MGT.
                                ;PREPARE PAGE ADDRESS REGISTERS FOR RELOCATION
024240  004737  026362                  JSR     PC,PREPAR               ;PREPARE MEM MGT FOR RELOCATION
                                ;INITIALIZE TEST PARAMETERS
024244  013765  005712  000002  9$:     MOV     FC,P.CYLN(R5)           ;INIT CYL TO FC.
024252  012701  000001          44$:    MOV     #1,R1                   ;INIT PATTERN BIT POINTER
024256  030137  005740          10$:    BIT     R1,PT                   ;SEE IF THIS PATTERN SELECTED
024262  001003                          BNE     14$                     ;BR IF THIS PATTERN IS SELECTED
024264  006301                  11$:    ASL     R1                      ;SHIFT PATTERN BIT POINTER
024266  001571                          BEQ     25$                     ;BR IF ALL PATTERNS CHECKED ON THIS CYLINDER
024270  000772                          BR      10$                     ;GO CHECK ANOTHER PATTERN
024272  113765  005720  000005  14$:    MOVB    FT,P.TRCK(R5)           ;INIT TRACK TO FT
024300  113765  005506  000004  16$:    MOVB    FS,P.SECT(R5)           ;INITIALIZE SECTOR TO FS
024306  013765  005746  000012  17$:    MOV     WC,P.WC(R5)             ;SET WORD COUNT FOR WC WORDS
024314  005465  000012                  NEG     P.WC(R5)
024320  004737  035330                  JSR     PC,CHKLIM               ;CHECK WRD CNT AND PACK ADR TO AVOID OVERFLOW
024324  024652                          25$                             ;RETURN ADDRESS FOR TRANSFER NOT ALLOWED
                                ;PERFORM TESTS AT CURRENT ADDRESS
024326  012737  024326  001110  18$:    MOV     #.,$LPERR               ;SET NEW LOOP ON ERROR ADRS
024334  004737  030032                  JSR     PC,SETTUP               ;SET UP FOR LOOP ON ERROR
024340  004737  037172                  JSR     PC,SVPRMS               ;STORE PARAMS FOR THIS XFER
024344  004737  034174                  JSR     PC,LODBUF               ;LOAD DATA INTO R/W BUFFER
024350  105037  003130                  CLRB    REISSU                  ;DUAL-ACC COMMAND RE-ISSUE FLAG
024354  032777  000040  154556          BIT     #BIT5,@SWR              ;SEE IF DATA WRITES INHIBITED
024362  001005                          BNE     20$                     ;BR IF INHIBITED
024364  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5)      ;SET WRITE COMMAND
024372  004737  037304                  JSR     PC,TRNSFR               ;WRITE THE DATA
024376  032777  000020  154534  20$:    BIT     #BIT4,@SWR              ;SEE IF WRITE CHECKS INHIBITED
024404  001014                          BNE     22$                     ;BR IF INHIBITED
024406  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5)      ;SET WRITE CHECK COMMAND
024414  004737  037304                  JSR     PC,TRNSFR               ;DO WRITE-CHECK OF DATA WRITTEN
024420  105737  003131                  TSTB    WCEFLG                  ;SEE IF A WCE ERROR OCCURRED
024424  001404                          BEQ     22$                     ;BR IF NOT
024426  032777  000002  154504          BIT     #BIT1,@SWR              ;SEE IF READ AND COMPARE SHOULD BE DONE
024434  001417                          BEQ     24$                     ;BR IF NOT
024436  032777  000010  154474  22$:    BIT     #BIT3,@SWR              ;SEE IF READ AND COMPARE INHIBITED
024444  001013                          BNE     24$                     ;BR IF INHIBITED
024446  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5)      ;SET READ COMMAND
024454  004737  037304                  JSR     PC,TRNSFR               ;READ THE DATA
024460  032777  000004  154452          BIT     #BIT2,@SWR              ;SEE IF SOFTWARE COMPARES INHIBITED
024466  001002                          BNE     24$                     ;BR IF INHIBITED
024470  004737  034422                  JSR     PC,CMPBUF               ;PERFORM SOFTWARE COMPARE OF DATA
024474  105737  003126          24$:    TSTB    DULACS                  ;SEE IF DUAL-ACCESS DATA TEST
024500  001405                          BEQ     27$                     ;BR IF NOT DUAL-ACCESS
024502  112765  000140  000001          MOVB    #RELEAS,P.CMND(R5)      ;SET DRIVE RELEASE COMMAND
024510  004737  037662                  JSR     PC,DRVCAL               ;RELEASE THE DRIVE
024514  104411                  27$:    SCOPER                          ;CHECK FOR INTERNAL LOOP ON ERROR
024516  005737  005740                  TST     PT                      ;SEE IF QUICK VERIFY DATA TEST DESIRED
024522  001031                          BNE     40$                     ;BR IF NOT QUICK VERIFY
                                ;INCREMENT PACK ADDRESS FOR QUICK VERIFY DEFAULT DATA TEST
024524  062765  000006  000004          ADD     #6,P.SECT(R5)           ;INCREMENT SECTOR BY 6
024532  126527  000004  000030          CMPB    P.SECT(R5),#24.         ;SEE IF SECTOR LIMIT REACHED YET
024540  002002                          BGE     34$                     ;BR IF LIMIT EXCEEDED
024542  000137  024160                  JMP     6$                      ;GO BACK AND TEST
024546  105265  000005          34$:    INCB    P.TRCK(R5)              ;INCREMENT TRACK
```

```
                                                ;SEE IF TRACK LIMIT EXCEEDED
                                                ;BR IF YES
                                                ;
                                                ;INCREMENT CYL
                                                ;SEE IF CYL LIMIT EXCEEDED
                                                ;BR IF YES
                                                ;
                                ;INCREMENT PACK ADDRESS FOR NON-DEFAULT DATA TEST
                                                ;INCREMENT SECTOR BY IS
                                                ;SEE IF LS EXCEEDED
                                                ;BR IF NOT YET
                                                ;GET TRACK NO.
                                                ;INCREMENT TRACK
                                                ;PUT IT BACK
                                                ;SEE IF LT EXCEEDED
                                                ;BR IF NOT YET
                                                ;JMP TO CHECK FOR NEXT PATTERN
                                                ;SEE IF FC>LC
                                                ;BR IF FC>LC
                                                ;INCREMENT THE CYLINDER
                                                ;SEE IF LC EXCEEDED
                                                ;BR IF LC EXCEEDED
                                                ;
                                                ;DECREMENT THE CYLINDER
                                                ;SEE IF LC EXCEEDED
                                                ;BR IF LC EXCEEDED
                                                ;
                                                ;SEE IF XXDP MIGHT BE OVERLAID NOW
                                                ;BR IF NOT
                                                ;RESTORE XXDP LOADER, IF SAVED
                                                ;SEE IF UNIBUS MAP PRESENT
                                                ;BR IF NOT
                                                ;DISABLE 22-BIT MODE AND UNIBUS MAP
32$:


.SBTTL  END OF PASS ROUTINE

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

$EOP:
        SCOPE
        INC     $DEVCT                  ;INCREMENT DEVICE COUNT FOR APT
        JMP     NEWDRV                  ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
DJNPAS: ;THIS IS TRULY THE END OF A PASS
        BIC     #BIT6,@$TKS             ;DISABLE TTY KBD INTERRUPT
        CLR     $TSTNM                  ;;ZERO THE TEST NUMBER
        CLR     $TIMES                  ;;ZERO THE NUMBER OF ITERATIONS
        INC     $PASS                   ;;INCREMENT THE PASS NUMBER
        BIC     #100000,$PASS           ;;DON'T ALLOW A NEG. NUMBER
        DEC     (PC)+                   ;;LOOP?
```

```
025016  000001                      $EOPCT: .WORD   1
025020  003022                              BGT     $DOAGN          ;;YES
025022  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
025024  000001                      $ENDCT: .WORD   1
025026  025016                              $EOPCT
025030  104401  025075                      TYPE    ,$ENDMG         ;;TYPE "END PASS #"
025034  013746  001326                      MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
025040  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
025042  104401  025072                      TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
025046  013700  000042              $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
025052  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
025054  000005                              RESET                   ;;CLEAR THE WORLD
025056  004710                      $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
025060  000240                              NOP                     ;;SAVE ROOM
025062  000240                              NOP                     ;;FOR
025064  000240                              NOP                     ;;ACT11
025066                              $DOAGN:
025066  000137                              JMP     @(PC)+          ;;RETURN
025070  016760                      $RTNAD: .WORD   NEWPAS
025072     377     377     000      $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
025075     015  042412  042116      $ENDMG: .ASCIZ  <15><12>/END PASS #/
025102  050040  051501  020123
025110  000043



                                    ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
                                    ;  WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
                                    ;      (NED = NON-EXISTENT DRIVE)
025112  032765  010000  000020      NEDHDL: BIT     #NED,P.CS2(R5)  ;SEE IF NED ON DRIVE SELECT
025120  001002                              BNE     1$              ;BR IF NED SET
025122  000137  041354                      JMP     ERRHDL          ;GO HANDLE  OTHER ERROR
025126  010046                      1$:     MOV     R0,-(SP)        ;SAVE R0,R1
025130  010146                              MOV     R1,-(SP)
025132  012700  000001                      MOV     #1,R0           ;SET BIT POINTER
025136  005001                              CLR     R1              ;CLEAR COUNTER
025140  120137  005500              2$:     CMPB    R1,DRIVE        ;SEE IF R1 = CURRENT DRIVE NUMBER
025144  001403                              BEQ     3$              ;BR IF  =
025146  005201                              INC     R1              ;INCREMENT COUNTER
025150  006300                              ASL     R0              ;SHIFT BIT POINTER
025152  000772                              BR      2$              ;TRY AGAIN
025154  040037  005530              3$:     BIC     R0,NEWON        ;CLEAR ONLINE BIT FOR  THIS DRIVE
025160  012601                              MOV     (SP)+,R1        ;RESTORE R0,R1
025162  012600                              MOV     (SP)+,R0
025164  000137  043570                      JMP     RETNML          ;TAKE  NORMAL  RETURN


                                    ;;*****************************************************************
                                    .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
                                    ;;*****************************************************************

025170  010146                      KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
025172  017701  153750                      MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
025176  042701  177600                      BIC     #177600,R1      ;MASK OUT UNUSED BITS
025202  120127  000172                      CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
025206  003005                              BGT     20$             ;BR IF NOT
```

```
6363  025310  120127  000141          CMPB    R1,#141
6364  025314  002402                  BLT     20$
6365  025316  042701  000040          BIC     #BIT5,R1        ;MAKE IT UPPER CASE
6366  025322  010137  005522          MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
6367  025326  122701  000003  20$:    CMPB    #3,R1           ;SEE IF (↑C) TYPED
6368  025332  001007                  BNE     2$              ;BR IF NOT (↑C)
6369  025334  104401  012362          TYPE    ,CNTRLC         ;ECHO (↑C)
6370  025340  042777  000100  153676  1$:     BIC     #BIT6,@#STKS    ;CLEAR KBD INTERRUPT ENABLE BIT
6371  025346  012601                  MOV     (SP)+,R1        ;RESTORE R1
6372  025350  000002                  RTI                     ;RETURN
6373  025352  122701  000032          2$:     CMPB    #032,R1         ;SEE IF (↑Z) TYPED
6374  025356  001003                  BNE     3$              ;BR IF NOT (↑Z)
6375  025360  104401  012367          TYPE    ,CNTRLZ         ;ECHO (↑Z)
6376  025364  000765                  BR      1$              ;RETURN
6377  025366  122701  000022          3$:     CMPB    #022,R1         ;SEE IF (↑R) TYPED
6378  025372  001003                  BNE     4$              ;BR IF NOT (↑R)
6379  025374  104401  012374          TYPE    ,CNTRLR         ;ECHO (↑R)
6380  025300  000757                  BR      1$              ;RETURN
6381  025302  122701  000007          4$:     CMPB    #007,R1         ;SEE IF (↑G) TYPED
6382  025306  001003                  BNE     5$              ;BR IF NOT (↑G)
6383  025310  104401  012406          TYPE    ,CNTRLG         ;ECHO (↑G)
6384  025314  000751                  BR      1$              ;RETURN
6385  025316  104401  005522          5$:     TYPE    ,INTCHR         ;ECHO INPUT
6386  025322  104401  001315          TYPE    ,$CRLF          ;DO <CR> AND <LF>
6387  025326  000744                  BR      1$              ;RETURN.
6388
6389
6390
6391                                  ;;*****************************************************************
6392                                  ;SBTTL  PREPKB - PREPARE FOR KEYBOARD INPUT
6393                                  ;*THIS SUBROUTINE CLEARS THE KEYBOARD BUFFER AND THE
6394                                  ;*DONE BIT, AND CLEARS THE TTY INTERRUPT INPUT WORD
6395                                  ;*INTCHR, IN PREPARATION FOR NEW TTY INPUT. IT ALSO
6396                                  ;*ENABLES KBD INTERRUPT.
6397                                  ;* CALL:
6398                                  ;*      JSR     PC,PREPKB
6399                                  ;;*****************************************************************
6400
6401  025330  005077  153612          PREPKB: CLR     @#STKB          ;CLEAR KBD BUFFER AND DONE BIT
6402  025334  005037  005522                  CLR     INTCHR          ;CLEAR TTY INPUT WORD
6403  025340  052777  000100  153576          BIS     #BIT6,@#STKS    ;ENABLE KBD INTERRUPT
6404  025346  000207                          RTS     PC              ;RETURN
6405
6406
6407
6408                                  ;;*****************************************************************
6409                                  ;SBTTL  ECOBAD - ECHO BAD TTY INPUT
6410                                  ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
6411                                  ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
6412                                  ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
6413                                  ;*AND <LF> ARE DONE.
6414                                  ;* CALL -     JSR     PC,ECOBAD
6415                                  ;;*****************************************************************
6416
6417  025350  104401  005522          ECOBAD: TYPE    ,INTCHR         ;ECHO BAD CHARACTER
6418  025354  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>, <LF>
```

```
 6419  025360  000207                        RTS     PC              ;RETURN
 6420
 6421
 6422
 6423                                ;;*****************************************************************
 6424                                ;*SIZMEM - SIZE MEMORY, SET LIMITS
 6425                                ;*THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE, AND
 6426                                ;*IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
 6427                                ;*IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D.
 6428                                ;*AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70.
 6429                                ;;*****************************************************************
 6430  025362  104427                SIZMEM: SAVREG                  ;SAVE R0-R5
 6431  025364  012737  000200  055134        MOV     #200,$KT11      ;SET MEM MGT KEY FOR $SIZE
 6432  025372  004737  055076                JSR     PC,$SIZE        ;SIZE MEMORY
 6433  025376  005737  055134                TST     $KT11           ;SEE IF MEM MGT PRESENT
 6434  025402  100406                        BMI     8$              ;BR IF MEM MGT PRESENT
 6435  025404  013737  055400  006106        MOV     $LSTAD,MAHILM   ;SET MEM LIMIT LO BITS
 6436  025412  005037  006110                CLR     MAHILM+2        ;CLEAR MEM LIMIT HI BITS
 6437  025416  000427                        BR      16$             ;GO TYPE LAST ADDRESS
 6438                                ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
 6439  025420  013700  055402        8$:     MOV     $LSTBK,R0       ;LO BITS
 6440  025424  005001                        CLR     R1              ;HI BITS
 6441  025426  012702  000006                MOV     #6,R2           ;SET LOOP COUNT = 6
 6442  025432  000241                12$:    CLC                     ;ROTATE LOOP
 6443  025434  006100                        ROL     R0
 6444  025436  006101                        ROL     R1
 6445  025440  005302                        DEC     R2
 6446  025442  001373                        BNE     12$
 6447                                ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
 6448  025444  063700  055400                ADD     $LSTAD,R0       ;ADD LO BITS
 6449  025450  005501                        ADC     R1              ;HI BITS
 6450  025452  010037  006106                MOV     R0,MAHILM       ;SET LO BITS OF MEM LIMIT
 6451  025456  010137  006110                MOV     R1,MAHILM+2     ;SET HI BITS OF MEM LIMIT
 6452  025462  042701  000003                BIC     #3,R1           ;CLEAR ADRS BITS 16,17
 6453  025466  001403                        BEQ     16$             ;BR IF NOT 22-BIT ADDRESSES
 6454  025470  112737  000001  003134        MOVB    #1,UBMPRS       ;SET "UNIBUS MAP PRESENT" FLAG
 6455                                ;TYPE "LAST PHYS MEM ADR = XXXXXXXX"
 6456  025476  104401  007303        16$:    TYPE    ,LSTMEM         ;TYPE "LAST PHYS MEM ADR ="
 6457  025502  012746  006106                MOV     #MAHILM,-(SP)   ;PUT POINTER ON STACK
 6458  025506  004737  052640                JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL ASCII
 6459  025512  004737  053154                JSR     PC,@#$SUPRS     ;TYPE "XXXXXXXX"
 6460  025516  104401  001315                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 6461  025522  104401  001315                TYPE    ,$CRLF
 6462  025526  104410                        RESREG                  ;RESTORE R0-R5
 6463  025530  000207                        RTS     PC              ;RETURN
 6464
 6465
 6466
 6467                                ;;*****************************************************************
 6468                                ;* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
 6469                                ;*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
 6470                                ;*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
 6471                                ;*TOP OF STACK.
 6472                                ;;*****************************************************************
 6473  025532  016646  000002        GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
 6474  025536  104403                        TYPOS                   ;TYPE IT
```

```
025540     006                            .BYTE   6                ;SIX DIGITS
025541     000                            .BYTE   0                ;SUPPRESS LEADING ZEROS
025542  104401  007511                    TYPE    ,NEWMSG          ;TYPE  "  NEW = "
025546  004737  030174                    JSR     PC,RDCHRS        ;READ NEW VALUE FROM KBD
025552  025604                            7$                       ;(↑C) RETURN ADDRESS
025554  025604                            7$                       ;(↑Z) RETURN ADDRESS
025556  025604                            7$                       ;(↑U) OR ERROR RETURN ADDRESS
025560  005700                            TST     R0               ;SEE IF ANY CHARS TYPED
025562  001001                            BNE     4$               ;BR IF YES
025564  000207                            RTS     PC               ;RETURN - OLD VALUE UNCHANGED
025566  020027  000006            4$:     CMP     R0,#6            ;SEE IF > 6 CHARS TYPED
025572  003407                            BLE     8$               ;BR IF NOT BAD
025574  104401  005262            6$:     TYPE    ,BUFFO           ;ECHO BAD INPUT
025600  104401  001314                    TYPE    ,$QUES
025604  162716  000010            7$:     SUB     #10,(SP)         ;FIX ERROR RETURN PC
025610  000207                            RTS     PC               ;ERROR RETURN
025612  012746  005262            8$:     MOV     #BUFFO,-(SP)     ;PUT POINTER TO CHARS ON STACK
025616  004737  051702                    JSR     PC,OCTBIN        ;CONVERT DIGITS TO BINARY
025622  025574                            6$                       ;ERROR RETURN ADDRESS
025624  012600                            MOV     (SP)+,R0         ;GET NEW BINARY VALUE
025626  005737  052034                    TST     $HIOCT           ;SEE IF HI BITS ARE 0
025632  001360                            BNE     6$               ;BR IF NOT
025634  010066  000002                    MOV     R0,2(SP)         ;PUT NEW VALUE ON STACK
025640  000207                            RTS     PC               ;RETURN



                                  ;;************************************************************
                                  .SBTTL  GTSWRG - OPEN SOFTWARE SWITCH REGISTER FOR MODIFICATION
                                  ;*THIS SUBROUTINE ALLOWS THE CONTENTS OF THE SOFTWARE SWITCH
                                  ;*REGISTER TO BE MODIFIED BY TTY INPUT. THE SUBROUTINE TYPES
                                  ;* "SWR = XXXXXX  NEW = ", AND WAITS FOR A NEW OCTAL VALUE
                                  ;*OF UP TO SIX DIGITS TO BE TYPED.
                                  ;;************************************************************
025642  022737  000176  001140    GTSWRG: CMP     #SWREG,SWR       ;SEE IF SOFTWARE SWR SELECTED
025650  001010                            BNE     6$               ;BR IF NOT
025652  013746  000176                    MOV     SWREG,-(SP)      ;PUT OLD VALUE ON STACK
025656  104401  007502                    TYPE    ,SWRMSG          ;TYPE "SWR = "
025662  004737  025532                    JSR     PC,GETPRM        ;TYPE OLD, GET NEW SWREG VALUE
025666  012637  000176                    MOV     (SP)+,SWREG      ;STORE NEW VALUE
025672  000207                    6$:     RTS     PC               ;RETURN



                                  ;;************************************************************
                                  ;*INITMM - INITIALIZE MEMORY MANAGEMENT REGISTERS
                                  ;*THIS SUBROUTINE INITIALIZES KIPAR'S AND KIPDR'S, FOR
                                  ;*CONTIGUOUS 4K PAGES, STARTING AT PHYSICAL ADDRESS 0.
                                  ;*KIPAR7 IS LOADED WITH 177600, TO PRESERVE THE I/O PAGE.
                                  ;*22-BIT MODE AND THE UNIBUS MAP ARE ENABLED, IF PRESENT,
                                  ;*BUT MEM MGT IS NOT TURNED ON IN THIS SUBROUTINE. HOWEVER,
                                  ;*MEM MGT TRAPS ARE ENABLED.

                                  ;;************************************************************
025674  104407                    INITMM: SAVREG                   ;SAVE R0-R5
025676  005001                            CLR     R1               ;INIT FOR PAR LOADING
```

```
6531  025700  012702  172340                MOV     #KIPAR0,R2      ;ADDR OF FIRST PAR
6532  025704  012703  000010                MOV     #8.,R3          ;LOAD 8 PAR'S AND 8 PDR'S
6533  025710  012762  077406  177740  4$:   MOV     #77406,-40.(R2) ;PDR = 4K UP,READ/WRITE
6534  025716  010122                        MOV     R1,(R2)+        ;LOAD A PAR
6535  025720  062701  000200                ADD     #200,R1         ;UPDATE FOR NEXT PAR
6536  025724  005303                        DEC     R3              ;DECREMENT LOOP COUNTER
6537  025726  001370                        BNE     4$              ;LOOP UNTIL ALL 8 ARE LOADED
6538  025730  012742  177600                MOV     #177600,-(R2)   ;SET UP KIPAR7 FOR I/O PAGE
6539  025734  105737  003134                TSTB    LBMPRS          ;SEE IF 22-BIT ADDRESSES
6540  025740  001403                        BEQ     10$             ;BR IF NOT
6541  025742  012737  000060  172516        MOV     #60,@#SR3       ;ENABLE 22-BIT MODE,AND UNIBUS MAP
6542  025750  012737  001000  177572  10$:  MOV     #BIT9,@#SR0     ;ENABLE KT11 TRAPS
6543  025756  104410                        RESREG                  ;RESTORE R0-R5
6544  025760  000207                        RTS     PC              ;RETURN
6545
6546
6547
6548                                  ;;*****************************************************************
6549                                  ;*SERVICE ROUTINE FOR MEM MGT TRAPS
6550                                  ;;*****************************************************************
6551  025762  004737  041074          KTERHD: JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
6552  025766  013737  177572  001174        MOV     @#SR0,$REG5     ;GET SR0
6553  025774  013737  177574  001176        MOV     @#SR1,$REG6     ;GET SR1
6554  026002  013737  177576  001200        MOV     @#SR2,$REG7     ;GET SR2
6555  026010  012737  026034  000004        MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
6556  026016  012737  000340  000006        MOV     #PR7,@#ERRVEC+2
6557  026024  013737  172516  001202        MOV     @#SR3,$REG10    ;GET SR3
6558  026032  000406                        BR      10$
6559  026034  022626                8$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
6560  026036  112737  000003  063500        MOVB    #3,DF30+18.     ;FIX PRINTOUT FOR NO SR3
6561  026044  105037  062141                CLRB    DH704+11.
6562  026050  104121                10$:    ERROR   121             ;KT11 FAILURE
6563  026052  000137  043364                JMP     HLTPRG          ;ABORT !!!
6564
6565
6566
6567                                  ;;*****************************************************************
6568                                  ;*ENBCSR - ENABLE MEMORY CSR'S FOR PARITY ERRORS
6569                                  ;;*****************************************************************
6570  026056  010146                ENBCSR: MOV     R1,-(SP)        ;SAVE R1
6571  026060  013746  000004                MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
6572  026064  013746  000006                MOV     @#ERRVEC+2,-(SP)
6573  026070  012737  026112  000004        MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
6574  026076  012701  172100                MOV     #MEMCSR,R1      ;ADRS OF MEMORY CSR'S
6575  026102  005011                4$:     CLR     (R1)            ;ZERO THIS CSR
6576  026104  012711  000001                MOV     #BIT0,(R1)      ;SET ENABLE IN THIS CSR
6577  026110  000401                        BR      10$
6578  026112  022626                8$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
6579  026114  062701  000002        10$:    ADD     #2,R1           ;INCREMENT TO NEXT CSR ADRS
6580  026120  020127  172140                CMP     R1,#MEMCSR+40   ;SEE IF DONE CHECKING YET
6581  026124  001366                        BNE     4$              ;BR IF NOT
6582  026126  012637  000006                MOV     (SP)+,@#ERRVEC+2 ;RESTORE OLD VECTORS
6583  026132  012637  000004                MOV     (SP)+,@#ERRVEC
6584  026136  012601                        MOV     (SP)+,R1        ;RESTORE R1
6585  026140  000207                        RTS     PC              ;RETURN
6586
```

```
                                        ;;*****************************************************************
                                        ;*SERVICE ROUTINE FOR MEM PARITY ERRORS
                                        ;;*****************************************************************
026142  010146                  MPERHD: MOV     R1,-(SP)        ;SAVE R1
026144  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE OLD VECTORS
026150  013746  000006                  MOV     @#ERRVEC+2,-(SP)
026154  004737  041074                  JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
026160  016637  000006  001174          MOV     6(SP),$REG5     ;GET PC OF ERROR
026166  012737  026312  000004          MOV     #10$,@#ERRVEC   ;SET T.O. VECTOR
026174  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
                        ;HANDLE 11/70 MEMORY PARITY ERROR
026202  013737  177740  001176          MOV     @#LOERAD,$REG6  ;LOW ERROR ADRS REG
026210  013737  177742  001200          MOV     @#HIERAD,$REG7  ;HI ERROR ADRS REG
026216  013737  177744  001202          MOV     @#MEMSYS,$REG10 ;MEMORY SYSTEM REG
026224  012737  062175  063522          MOV     #DH707,DF31+16. ;FIX ERROR MSG FOR 11/70
026232  112737  000004  063524          MOVB    #4,DF31+18.
026240  104122                          ERROR   122             ;11/70 MEM PARITY ERROR
026242  005737  001200                  TST     $REG7           ;SEE IF BAD MEMORY IS IN PROGRAM AREA
026246  001011                          BNE     6$              ;BR IF NOT
026250  023727  001176  071526          CMP     $REG6,#RWBUF+6000 ;SEE IF BAD MEM IS IN PROG. AREA
026256  103005                          BHIS    6$              ;BR IF NOT
026260  105737  003135          4$:     TSTB    MEMABT          ;SEE IF ABORT DESIRED
026264  001002                          BNE     6$              ;BR IF NOT
026266  000137  043364                  JMP     HLTPRG          ;ABORT !!!
026272  012637  000006          6$:     MOV     (SP)+,@#ERRVEC+2 ;RESTORE T.O. VECTOR
026276  012637  000004                  MOV     (SP)+,@#ERRVEC
026302  012601                          MOV     (SP)+,R1        ;RESTORE R1
026304  004737  026056                  JSR     PC,ENBCSR       ;GO CLEAR AND ENABLE CSR'S
026310  000002                          RTI                     ;RETURN
                        ;HANDLE ALL OTHER MEMORY PARITY ERRORS  (NON-11/70)
026312  022626          10$:    CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
026314  012737  026344  000004          MOV     #14$,@#ERRVEC   ;SET T.O. VECTOR
026322  012701  172100                  MOV     #MEMCSR,R1      ;GET FIRST CSR ADDRESS
026326  011137  001200          12$:    MOV     (R1),$REG7      ;CHECK FOR A MEMORY CSR
026332  100005                          BPL     16$             ;BR IF NO ERROR SET HERE
026334  010137  001176                  MOV     R1,$REG6        ;GET CSR ADDRESS
026340  104122                          ERROR   122             ;MEMORY PARITY ERROR (NON-11/70)
026342  000746                          BR      4$              ;GO SEE IF SHOULD ABORT
026344  022626          14$:    CMP     (SP)+,(SP)+     ;CLEAN UP STACK
026346  062701  000002          16$:    ADD     #2,R1           ;INCR R1 TO POINT TO NEXT CSR
026352  020127  172140                  CMP     R1,#MEMCSR+40   ;SEE IF DONE CHECKING
026356  103763                          BLO     12$             ;BR IF NOT
026360  000744                          BR      6$              ;RETURN


                                        ;;*****************************************************************
                                        ;*PREPAR - PREPARE MEM MGT FOR RELOCATION
                                        ;*THIS SUBROUTINE CALLS INITMM, AND THEN SETS UP CONSTANT
                                        ;*FOR KIPAR6 IN SAVPAR, AND LOADS THE UNIBUS MAP REGISTERS.
                                        ;*IF PRESENT.
                                        ;;*****************************************************************
026362  104407          PREPAR: SAVREG                  ;SAVE R0-R5
026364  004737  025674                  JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
```

```
6643  026370  013700  005572                  MOV     PMA,R0          ;LO BITS OF MA
6644  026374  042700  017777                  BIC     #17777,R0       ;MASK FOR BITS 13-15
6645  026400  013701  005574                  MOV     PMA+2,R1        ;HI BITS OF MA
6646  026404  010003                          MOV     R0,R3           ;SAVE THESE BITS
6647  026406  010104                          MOV     R1,R4
6648  026410  006100                          ROL     R0              ;GET MA BITS 13-21 INTO R1 BITS 7-15
6649  026412  006101                          ROL     R1
6650  026414  006100                          ROL     R0
6651  026416  006101                          ROL     R1
6652  026420  000301                          SWAB    R1
6653  026422  006100                          ROL     R0
6654  026424  106001                          RORB    R1
6655  026426  010137  003176                  MOV     R1,SAVPAR       ;CONSTANT FOR LOADING PAR6 LATER
6656  026432  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP ENABLED
6657  026436  001420                          BEQ     9$              ;BR IF NOT (NO UNIBUS MAPPING)
6658                                  ;PREPARE FOR 22-BIT ADDRESSING -  LOAD UNIBUS MAP REGISTERS
6659  026440  012700  170200                  MOV     #MAPLOO,R0      ;STARTING ADDR OF MAP REGISTERS
6660  026444  012701  000037                  MOV     #31.,R1         ;SET REGISTER COUNTER
6661  026450  010320          4$:     MOV     R3,(R0)+        ;LOAD A MAP REGISTER
6662  026452  010420                  MOV     R4,(R0)+
6663  026454  062703  020000                  ADD     #20000,R3       ;ADD 4K WORDS
6664  026460  005504                          ADC     R4
6665  026462  077106                          SOB     R1,4$           ;LOOP UNTIL 31(DEC) ARE LOADED
6666  026464  042765  160000  000010          BIC     #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
6667  026472  142765  000003  000007          BICB    #3,P.BAHI(R5)
6668  026500  104410          9$:     RESREG                  ;RESTORE P0-R5
6669  026502  000207                          RTS     PC              ;RETURN
6670
6671
6672
6673                                  ;;*****************************************************************
6674                                  ;* FNDXDP - FIND STARTING ADR OF XXDP LOADER , AND STORE IT
6675                                  ;*IN XXDPAD.
6676                                  ;;*****************************************************************
6677  026504  005737  006110  FNDXDP: TST     MAHILM+2        ;TEST HI BITS OF UPPER MEMORY LIMIT
6678  026510  001404                  BEQ     6$              ;BR IF HI BITS ARE 0
6679  026512  012737  160000  005536  4$:     MOV     #160000,XXDPAD  ;START OF 28K IS END OF XXDP
6680  026520  000412                  BR      8$
6681  026522  023727  006106  157776  6$:     CMP     MAHILM,#157776  ;SEE IF MEM LIMIT > OR = 157776
6682  026530  103370                  BHIS    4$              ;BR IF YES
6683  026532  013737  006106  005536          MOV     MAHILM,XXDPAD   ;HI MEM LIMIT IS END OF XXDP
6684  026540  062737  000002  005536          ADD     #2,XXDPAD       ;ADD 2 BYTES
6685  026546  162737  006000  005536  8$:     SUB     #6000,XXDPAD    ;COMPUTE START OF XXDP
6686  026554  000207                  RTS     PC              ;RETURN
6687
6688
6689
6690                                  ;;*****************************************************************
6691                                  ;*CHKXDP - CHECK FOR IMMINENT OVERLAY OF XXDP LOADER
6692                                  ;*THIS SUBROUTINE DETERMINES IF THE CURRENT MA AND WC COMBINATION
6693                                  ;*WOULD CAUSE OVERLAY OF THE XXDP LOADER ON A WRITE TRANSFER.
6694                                  ;*IF SO, AN ATTEMPT IS MADE TO MOVE THE LOADER INTO UNUSED MEMORY.
6695                                  ;*IF THIS IS NOT POSSIBLE, A RETURN IS MADE TO THE ADDRESS
6696                                  ;*FOLLOWING THE CALL. THE SAVE ADDRESS IS STORED IN XXDPSAV AND XXDPSAV+2.
6697                                  ;*IF IT IS NOT NECESSARY TO MOVE THE LOADER, A NORMAL RETURN IS MADE.
6698                                  ;* CALL  -       JSR     PC,CHKXDP
```

```
                                        ;*              <RETURN ADR FOR NO SAVE>
                                        ;;*****************************************************
026556  104407                          CHKXDP: SAVREG                          ;SAVE R0-R5
                                        ;SEE IF LOADER MUST BE PROTECTED
026560  105737  003126                          TSTB    DULACS                  ;SEE IF DUAL-ACCESS DATA TEST
026564  001005                                  BNE     2$                      ;BR IF YES
026566  012700  005606                          MOV     #TSTLST,R0              ;GET ADRS OF TEST LIST
026572  005760  000040                          TST     RWTINX(R0)              ;SEE IF R/W DATA TEST TO BE RUN
026576  001406                                  BEQ     4$                      ;EXIT IF NOT TO BE RUN
026600  005737  005740                  2$:     TST     PT                      ;SEE IF DEFAULT DATA TEST
026604  001403                                  BEQ     4$                      ;BR IF DEFAULT DATA TEST
026606  005737  005744                          TST     MA+2                    ;SEE IF HI MEM ADR = 0
026612  001404                                  BEQ     6$                      ;BR IF 0
026614  062716  000002                  4$:     ADD     #2,(SP)                 ;FIX UP NORMAL RETURN PC
026620  104410                          5$:     RESREG                          ;RESTORE R0-R5
026622  000207                                  RTS     PC                      ;RETURN
026624  013700  005746                  6$:     MOV     WC,R0                   ;GET WORD COUNT LO BITS
026630  005300                                  DEC     R0                      ;DECREMENT IT
026632  005001                                  CLR     R1                      ;SET HI BITS = 0
026634  000241                                  CLC                             ;DOUBLE IT FOR BYTES
026636  006100                                  ROL     R0
026640  006101                                  ROL     R1
026642  063700  005742                          ADD     MA,R0                   ;COMPUTE LAST ADR OF XFER IN R1-R0
026646  005501                                  ADC     R1
026650  063701  005744                          ADD     MA+2,R1
026654  105037  003125                          CLRB    XDPSVD                  ;INIT LOADER SAVE INDICATOR
026660  004737  026504                          JSR     PC,FNDXDP               ;COMPUTE START ADR OF XXDP
026664  013703  005536                          MOV     XXDPAD,R3               ;GET START OF XXDP LOADER
026670  062703  005776                          ADD     #5776,R3                ;COMPUTE LAST ADDRESS OF XXDP
026674  023703  005742                          CMP     MA,R3                   ;SEE IF MA > LAST ADR OF XXDP
026700  101345                                  BHI     4$                      ;BR IF YES - DON'T MOVE XXDP
026702  005701                                  TST     R1                      ;CHECK HI BITS OF LAST XFER ADDRESS
026704  001003                                  BNE     8$                      ;BR IF NOT 0
026706  020037  005536                          CMP     R0,XXDPAD               ;CHECK LO BITS OF LAST XFER ADR
026712  103740                                  BLO     4$                      ;BR IF < XXDP ADR - DON'T MOVE XXDP
026714  105237  003125                  8$:     INCB    XDPSVD                  ;SET SAVE INDICATOR
                                        ;ATTEMPT TO FIND A SAVE AREA FOR LOADER
026720  023727  005742  100000                  CMP     MA,#100000              ;SEE IF MA > OR = 16K
026726  103410                                  BLO     12$                     ;BR IF NOT
026730  012737  072000  005540                  MOV     #72000,XDPSAV           ;SAVE ADR = 72000
026736  005037  005542                          CLR     XDPSAV+2
026742  004737  027032                  10$:    JSR     PC,SAVXDP               ;SAVE THE LOADER
026746  000722                                  BR      4$                      ;BR TO RETURN
026750  062700  000002                  12$:    ADD     #2,R0                   ;ADD 2 BYTES TO LAST ADR OF XFER
026754  005501                                  ADC     R1
026756  010002                                  MOV     R0,R2                   ;GET A COPY
026760  010103                                  MOV     R1,R3
026762  062702  005776                          ADD     #5776,R2                ;COMPUTE END OF SAVE AREA
026766  005503                                  ADC     R3
026770  020337  006110                          CMP     R3,MAHILM+2             ;SEE IF THIS EXCEEDS MEMORY LIMIT
026774  101011                                  BHI     16$                     ;BR IF YES - CAN'T SAVE LOADER
026776  001003                                  BNE     14$                     ;BR IF < MEM LIMIT
027000  020237  006106                          CMP     R2,MAHILM               ;SEE IF LO BITS EXCEED MEM LIMIT
027004  101005                                  BHI     16$                     ;BR IF YES - CAN'T SAVE LOADER
027006  010037  005540                  14$:    MOV     R0,XDPSAV               ;SET SAVE ADR IN HI MEMORY
027012  010137  005542                          MOV     R1,XDPSAV+2
```

```
                        000751                          BR      10$             ;GO SAVE LOADER
                        017616  000000          16$:    MOV     2(SP),(SP)      ;FIX UP NO-SAVE RETURN PC
                        105037  003125                  CLRB    XDPSVD          ;CLEAR THE SAVE INDICATOR
                        000673                          BR      5$              ;BR TO RETURN


                                                ;;*****************************************************************
                                                ;* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
                                                ;*THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
                                                ;*PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
                                                ;*STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
                                                ;*
                                                ;* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
                                                ;*THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
                                                ;*STORED IN XDPSAV,XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
                                                ;*ADDRESS).
                                                ;;*****************************************************************
                        104407          SAVXDP: SAVREG                          ;SAVE R0-R5
                        005004                  CLR     R4                      ;SET INDICATOR TO SAVE XXDP
                        000410                  BR      XDP1
                        104407          GETXDP: SAVREG                          ;SAVE R0-R5
                        105037  003124          CLRB    XOVLAD                  ;CLEAR XXDP OVERLAID INDICATOR
                        105737  003125          TSTB    XDPSVD                  ;SEE IF XXDP WAS SAVED
                        001425                  BEQ     XDP2                    ;BR IF NOT SAVED
                        012704  000001          MOV     #1,R4                   ;SET INDICATOR TO GET XXDP
                        105737  000041  XDP1:   TSTB    @#41                    ;SEE IF LOADED BY XXDP
                        001420                  BEQ     XDP2                    ;BR IF NOT
                        012702  003000          MOV     #1536.,R2               ;GET SET TO MOVE 1536(DEC) WORDS
                        013703  005536          MOV     XXDPAD,R3               ;GET ORIG ADR OF START OF XXDP
                        013700  005540          MOV     XDPSAV,R0               ;GET SAVE ADR LO BITS
                        013701  005542          MOV     XDPSAV+2,R1             ;HI BITS
                        005737  055134          TST     $KT11                   ;SEE IF MEM MGT PRESENT
                        100417                  BMI     XDP3                    ;BR IF PRESENT
                        005704                  TST     R4                      ;SEE IF WANT TO SAVE OR GET XXDP
                        001005                  BNE     XDP4                    ;BR IF WANT TO GET XXDP
                        012320          6$:     MOV     (R3)+,(R0)+             ;SAVE A WORD
                        005302                  DEC     R2                      ;SEE IF 1536(DEC) WORDS YET
                        001375                  BNE     6$                      ;BR IF NOT YET
                        104410          XDP2:   REGREG                          ;RESTORE R0-R5
                        000207                  RTS     PC                      ;RETURN
                        062700  006000  XDP4:   ADD     #6000,R0                ;POINT TO END OF SAVE AREA
                        062703  006000          ADD     #6000,R3                ;POINT TO END OF XXDP LOADER AREA
                        014043          8$:     MOV     -(R0),-(R3)             ;GET A WORD
                        005302                  DEC     R2                      ;SEE IF 1536(DEC) WORDS YET
                        001375                  BNE     8$                      ;BR IF NOT YET
                        000766                  BR      XDP2                    ;GO EXIT
                                        ;COME HERE IF MEM MGT
                        004737  025674  XDP3:   JSR     PC,INITMM               ;INIT MEM MGT REGISTERS
                        006100                  ROL     R0                      ;GET ADR BITS 13-21 INTO R1 BITS 7-15
                        006101                  ROL     R1
                        006100                  ROL     R0
                        006101                  ROL     R1
                        000301                  SWAB    R1
                        006100                  ROL     R0
                        106001                  RORB    R1
```

```
6811  027174  010137  172354                  MOV     R1,@#KIPAR6     ;SET UP PAR6
6812  027200  013701  005540                  MOV     XDPSAV,R1       ;GET LO ADRS BITS
6813  027204  042701  160000                  BIC     #160000,R1      ;FIX UP R1 TO REFERENCE PAR6
6814  027210  052701  140000                  BIS     #140000,R1
6815  027214  005704                  16$     TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
6816  027216  001007                          BNE     18$             ;BR IF WANT TO GET XXDP
6817  027220  012305                          MOV     (R3)+,R5        ;SAVE A WORD
6818  027222  005237  177572                  INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
6819  027226  010521                          MOV     R5,(R1)+
6820  027230  005337  177572                  DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
6821  027234  000406                          BR      20$
6822  027236  005237  177572          18$:    INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
6823  027242  012105                          MOV     (R1)+,R5        ;GET A WORD
6824  027244  005337  177572                  DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
6825  027250  010523                          MOV     R5,(R3)+
6826  027252  032701  020000          20$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO PAGE 7
6827  027256  001405                          BEQ     22$             ;BR IF NOT
6828  027260  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
6829  027264  062737  000200  172354          ADD     #200,@#KIPAR6   ;UPDATE PAR BY 4K
6830  027272  005302                  22$:    DEC     R2              ;SEE IF 1536 WORDS YET
6831  027274  001347                          BNE     16$             ;BR IF NOT YET
6832  027276  000713                          BR      XDP2            ;BR TO RETURN
6833
6834
6835
6836                                  ;;*****************************************************************
6837                                  .SBTTL  INITSS - INITIALIZE SUBSYSTEM
6838                                  ;*
6839                                  ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
6840                                  ;*AND DOES A SUBSYSTEM CLEAR.
6841                                  ;* USES -   R2,R5
6842                                  ;* CALL:
6843                                  ;*              JSR     PC,INITSS
6844                                  ;;*****************************************************************
6845
6846  027300  012737  041354  003036  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
6847  027306  012737  040106  003034          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
6848  027314  013702  003026                  MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
6849  027320  012705  002620                  MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
6850  027324  105037  003133                  CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
6851  027330  004737  027372                  JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
6852  027334  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6853  027342  004737  037662                  JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
6854  027346  113765  005500  000000          MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
6855  027354  113737  005500  002704          MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
6856  027362  113765  003115  000007          MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
6857  027370  000207                          RTS     PC              ;SUBROUTINE EXIT
6858
6859
6860
6861                                  ;;*****************************************************************
6862                                  .SBTTL  CLRPRM - CLEAR DRIVER INPUT PARAMETERS
6863                                  ;*THIS SUBROUTINE ZEROS THE FIRST 14 BYTES IN THE DRIVER
6864                                  ;*PARAMETER BLOCK (WHOSE ADDRESS IS IN R5).
6865                                  ;* CALL -       JSR     PC,CLRPRM
6866                                  ;;*****************************************************************
```

```
                                ;CLRPRM - CLEAR DRIVER INPUT PARAMETERS

027372  010046          CLRPRM: MOV     R0,-(SP)        ;SAVE R0
027374  010546                  MOV     R5,-(SP)        ;SAVE R5
027376  010500                  MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
027400  062705  000016          ADD     #P.CS1,R5       ;COMPUTE LIMIT ADDRESS
027404  005020          1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
027406  020005                  CMP     R0,R5           ;SEE IF DONE YET
027410  001375                  BNE     1$              ;BR IF  NOT DONE  YET
027412  012605                  MOV     (SP)+,R5        ;RESTORE R5
027414  012600                  MOV     (SP)+,R0        ;RESTORE R0
027416  000207                  RTS     PC              ;RETURN


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

027420  004737  027300  SCNDRV: JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
027424  113765  005500  000000  MOVB    DRIVE,P.DRVN(R5) ;GET DRIVE NUMBER
                                ;SEE IF DRIVE 0 IS XXDP LOAD MEDIUM
027432  001012                  BNE     3$              ;BR IF NOT DRIVE 0
027434  122737  000013  000041  CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
027442  001006                  BNE     3$              ;BR IF NOT
027444  105737  003106          TSTB    MDFLAG          ;SEE IF 200 START
027450  001003                  BNE     3$              ;BR IF NOT
027452  104401  010045          TYPE    ,DROXDP         ;TYPE "DRIVE 0 IS LOAD MEDIUM"
027456  000434                  BR      2$              ;TAKE ERROR EXIT
027460  112765  000141  000001  3$:     MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
027466  012737  025112  003036          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
027474  012737  000377  005530          MOV     #377,NEWON      ;INIT. ON-LINE INDICATOR
027502  004737  037662                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
                                ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
027506  012737  041354  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNL RETURN ADDRESS
027514  022737  000377  005530          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
027522  001420                  BEQ     4$              ;BR IF NED NOT SET
027524  113737  005500  007546          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
027532  152737  000060  007546          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
027540  104401  007540                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
027544  104401  007551                  TYPE    ,NXDRIV         ;TYPE "NON-EXISTENT"
                                ;SERVICE ERRORS HERE
027550  042762  000100  000000  2$:     BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
027556  017616  000000                  MOV     @(SP),(SP)      ;SET UP ERROR RETURN ADDRESS
```

```
                                        RTS     PC                      ;ERROR RETURN
                                ;SEE IF DRIVE IS READY
000200  000040  4$:             BIT     #S.DRY,P.ADD(R5) ;TEST FOR DRIVE READY
                                BNE     6$              ;BR IF DRIVE IS READY
                                MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
005500  007546                  BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
                                TYPE    ,BADDRV         ;TYPE "DRIVE X"
                                TYPE    ,NOTRDY         ;TYPE "NOT READY"
                                BR      2$              ;TAKE ERROR EXIT
                                ;SEE IF DRIVE IS WRITE ENABLED
004000  000040  6$:             BIT     #S.WRL,P.ADD(R5 ;SEE IF WRITE LOCK SET
                                BEQ     8$              ;BR IF WRITE LOCK NOT SET
005500  007546                  MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
                                BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
                                TYPE    ,BADDRV         ;TYPE "DRIVE X"
                                TYPE    ,WPTLOK         ;TYPE "WRITE-LOCKED"
                                BR      2$              ;TAKE ERROR EXIT
                                ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
000103  000001  8$:             MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
                                JSR     PC,DRVCAL       ;SET VOLUME VALID
000001                          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
000632  000002                  MOV     #LSTCYL,P.CYLN(R5) ;SET CYLINDER = 632(8)
000002  000005                  MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
063526  000010                  MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
177774  000012                  MOV     #-4,P.WC(R5)    ;READ 4 WORDS
000007                          CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
                                JSR     PC,DRVCAL       ;READ 4 WORDS OF BAD SECTOR FILE
005474                          BIT     #ANYDER,RECODE  ;SEE IF DATA ERROR
                                BEQ     10$             ;BR IF OK
012316                          TYPE    ,BAD632         ;TYPE READ ERROR MESSAGE
177777  063534  10$:            CMP     #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
                                BNE     12$             ;BR IF NOT ALL 1'S
005500  007546                  MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
000060  007546                  BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
                                TYPE    ,BADDRV         ;TYPE "DRIVE X"
                                TYPE    ,ALNPAK         ;TYPE "LOADED WITH ALIGN PACK"
                                BR      2$              ;TAKE ERROR EXIT
                                ;ERROR FREE RETURN
000113  000001  12$:            MOVB    #RECAL,P.CMND(R5) ;SET RECALIBRATE COMMAND
                                JSR     PC,DRVCAL       ;RECALIBRATE THIS DRIVE
000002                          ADD     #2,(SP)         ;FIX UP RETURN PC
                                RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*SETTUP - SET UP FOR LOOP ON ERROR
                                ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
                                ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
                                ;;*****************************************************************
001076          SETTUP:         MOV     (SP),2#STACK-2  ;MOVE RETURN PC ON STACK
001076                          MOV     #STACK-2,SP     ;RE-INIT THE STACK POINTER
005474                          CLR     RECODE          ;CLEAR ERROR RECOVERY FLAGS
003116                          CLRB    ERRCNT          ;CLEAR RETRY ERROR COUNT
002620                          MOV     #PARMD,R5       ;SET PARAM BLK ADRS
000177  000001                  MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
```

```
030064  004737  037662                   JSR     PC,DRVCAL         ;CLEAR THE SUBSYSTEM
030070  012737  000000  177776           MOV     #PR0,@#PS         ;RE-ESTABLISH PRIORITY 0
030076  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.

030100  010146                   CHKITR: MOV     R1,-(SP)          ;SAVE R1
030102  112737  000144  001115           MOVB    #100.,$ERMAX      ;SET MAX ERROR CNT TO 100 FOR $SCOPE
030110  105737  003126                   TSTB    DULACS            ;SEE IF DUAL-ACCESS FLAG SET
030114  001006                           BNE     3$                ;BR IF YES
030116  105737  003106                   TSTB    MDFLAG            ;SEE IF 200 START
030122  001007                           BNE     4$                ;BR IF NOT
030124  005737  001326                   TST     $PASS             ;SEE IF FIRST PASS
030130  001004                           BNE     4$                ;BR IF NOT
030132  012737  000001  001304   3$:     MOV     #1,$TIMES         ;SET UP FOR 1 ITERATION
030140  000410                           BR      1$                ;GO RUN DUAL-ACCESS TEST
030142  113701  001102           4$:     MOVB    $TSTNM,R1         ;GET CURRENT TEST NUMBER
030146  005301                           DEC     R1                ;DECREMENT BY 1
030150  006301                           ASL     R1                ;DOUBLE IT, TO GET TEST LIST INDEX
030152  016137  005606  001304           MOV     TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
030160  001403                           BEQ     2$                ;BR IF TEST SHOULD BE SKIPPED
030162  062766  000004  000002   1$:     ADD     #4,2(SP)          ;ADJUST RETURN PC TO RUN THIS TEST
030170  012601                   2$:     MOV     (SP)+,R1          ;RESTORE R1
030172  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
                                        ;;*******************************************************************

        030174                          RDCHRS:
        030174  010146                          MOV     R1,-(SP)                ;SAVE R1
        030176  010246                          MOV     R2,-(SP)                ;SAVE R2
        030200  005000                          CLR     R0                      ;INITIALIZE CHARACTER COUNT
        030202  005001                          CLR     R1                      ;INITIALIZE RUB-OUT INDICATOR
                                        ;READ A CHARACTER
        030204  104406                  2$:     RDCHR                           ;READ A CHARACTER
        030206  112602                          MOVB    (SP)+,R2                ;GET CHARACTER INTO R2
                                        ;CHECK FOR (↑C)
        030210  122702  000003                  CMPB    #003,R2                 ;SEE IF (↑C) TYPED
        030214  001006                          BNE     4$                      ;BR IF NOT (↑C)
        030216  104401  012362                  TYPE    ,CNTRLC                 ;ECHO (↑C)
        030222  017666  000004  000004  3$:     MOV     @4(SP),4(SP)            ;PUT RETURN ADDRESS ON STACK
        030230  000523                          BR      24$                     ;BR TO TAKE EXIT
                                        ;CHECK FOR (↑Z)
        030232  122702  000032          4$:     CMPB    #032,R2                 ;SEE IF (↑Z) TYPED
        030236  001006                          BNE     6$                      ;BR IF NOT (↑Z)
        030240  104401  012367                  TYPE    ,CNTRLZ                 ;ECHO (↑Z)
        030244  062766  000002  000004          ADD     #2,4(SP)                ;MAKE OLD PC POINT TO NEXT RETURN ADR.
        030252  000763                          BR      3$                      ;BR TO TAKE (↑Z) EXIT
                                        ;CHECK FOR (↑U)
        030254  122702  000025          6$:     CMPB    #025,R2                 ;SEE IF (↑U) TYPED
        030260  001006                          BNE     8$                      ;BR IF NOT (↑U)
        030262  104401  012401                  TYPE    ,CNTRLU                 ;ECHO (↑U)
        030266  062766  000004  000004  7$:     ADD     #4,4(SP)                ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
        030274  000752                          BR      3$                      ;BR TO TAKE (↑U) EXIT
                                        ;CHECK FOR (↑G)
        030276  122702  000007          8$:     CMPB    #007,R2                 ;SEE IF (↑G) TYPED
        030302  001005                          BNE     9$                      ;BR IF NOT (↑G)
        030304  104401  012406                  TYPE    ,CNTRLG                 ;ECHO (↑G)
        030310  004737  025642                  JSR     PC,GTSWRG               ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
        030314  000764                          BR      7$                      ;TAKE (↑U) RETURN
                                        ;CHECK FOR RUB-OUT (DELETE)
        030316  122702  000177          9$:     CMPB    #177,R2                 ;SEE IF RUB-OUT (DEL) TYPED
        030322  001020                          BNE     14$                     ;BR IF NOT RUB-OUT
        030324  005700                          TST     R0                      ;CHECK THE CHARACTER COUNT
        030326  001726                          BEQ     2$                      ;BR IF COUNT = 0
        030330  005701                          TST     R1                      ;CHECK THE RUB-OUT INDICATOR
        030332  001003                          BNE     11$                     ;BR IF WE HAD A PREVIOUS RUB-OUT
        030334  005201                          INC     R1                      ;SET RUB-OUT INDICATOR
        030336  104401  012417                  TYPE    ,BKSLSH                 ;TYPE A BACK-SLASH (\)
        030342  005037  005532          11$:    CLR     SCRACH                  ;USE SCRATCH WORD FOR TEMP. BUFFER
        030346  005300                          DEC     R0                      ;DECREMENT COUNT
        030350  116037  005262  005532          MOVB    BUFF0(R0),SCRACH        ;GET LAST CHAR. INTO BUFFER
        030356  104401  005532                  TYPE    ,SCRACH                 ;ECHO CHARACTER TO BE DELETED
        030362  000710                          BR      2$                      ;GO READ ANOTHER CHARACTER
        030364  005701                  14$:    TST     R1                      ;CHECK THE RUB-OUT INDICATOR
        030366  001403                          BEQ     16$                     ;BR IF INDICATOR IS NOT SET
        030370  104401  012417                  TYPE    ,BKSLSH                 ;TYPE A BACK-SLASH
        030374  005001                          CLR     R1                      ;CLEAR THE RUB-OUT INDICATOR
                                        ;CHECK FOR CARRIAGE RETURN
        030376  122702  000015          16$:    CMPB    #015,R2                 ;SEE IF <CR> TYPED
```

```
030402  001426                          BEQ     19$                     ;BR IF <CR>
                                ;HANDLE POSSIBLE DIGIT
030404  005037  005532                  CLR     SCRACH                  ;USE SCRATCH WORD FOR TEMP. BUFFER
030410  110237  005532                  MOVB    R2,SCRACH               ;GET THIS CHARACTER INTO BUFFER
030414  104401  005532                  TYPE    ,SCRACH                 ;ECHO THE CHARACTER TYPED
030420  110260  005262                  MOVB    R2,BUFFO(R0)            ;PUT CHARACTER INTO BUFFER
030424  005200                          INC     R0                      ;INCREMENT CHARACTER COUNTER
030426  022700  000120                  CMP     #80.,R0                 ;SEE IF TOO MANY CHARACTERS TYPED
030432  001264                          BNE     2$                      ;BR IF NOT TOO MANY
030434  104401  001315                  TYPE    ,$CRLF                  ;TYPE <CR> AND <LF>
030440  112760  000000  005262          MOVB    #0,BUFFO(R0)          ;PUT TERMINATING NULL INTO BUFFER
030446  104401  005262                  TYPE    ,BUFFO                  ;ECHO INPUT STRING
030452  104401  001314                  TYPE    ,$QUES                  ;TYPE <?>,<CR>,<LF>
030456  000703                          BR      ^$                      ;TAKE ERROR EXIT FROM RDCHRS
030460  104401  001315          19$:    TYPE    ,$CRLF                  ;TYPE <CR>,<LF>
030464  112760  000000  005262          MOVB    #0,BUFFO(R0)            ;PUT TERMINATING NULL INTO BUFFER
030472  062766  000006  000004          ADD     #6,4(SP)                ;FIX UP RETURN ADDRESS
030500  012602                  24$:    MOV     (SP)+,R2                ;RESTORE R2
030502  012601                          MOV     (SP)+,R1                ;RESTORE R1
030504  000207                          RTS     PC                      ;SUBROUTINE EXIT



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

030506  104401  012430          TYPTST: TYPE    ,SPACE1                 ;TYPE A SPACE
030512  010146                          MOV     R1,-(SP)                ;PUT INDEX ONTO STACK
030514  006216                          ASR     (SP)                    ;DIVIDE BY 2
030516  005216                          INC     (SP)                    ;INCREMENT TO GET TEST NO.
030520  104403                          TYPOS                           ;TYPE TEST NO. IN OCTAL
030522     002                          .BYTE   2                       ;TYPE 2 DIGITS
030523     000                          .BYTE   0                       ;SUPPRESS LEADING ZEROS
030524  104401  012423                  TYPE    ,SPACE6                 ;TYPE 6 SPACES
030530  016146  005606                  MOV     $TLST(R1),-(SP)         ;PUT CURRENT ITERATION NO. ONTO STACK
030534  104403                          TYPOS                           ;TYPE ITERATION NO. IN OCTAL
030536     006                          .BYTE   6                       ;TYPE 6 DIGITS
030537     000                          .BYTE   0                       ;SUPPRESS LEADING ZEROS
030540  000207                          RTS     PC                      ;RETURN



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

```
                                        ;* CALL -      JSR     PC,TYPPRM
                                        ;;*****************************************************************

030542  016137  006132  012432    TYPPRM: MOV     PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
030550  104401  012432                    TYPE    ,PRMBUF           ;TYPE "XX="
030554  016137  005712  005466            MOV     PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
030562  005037  005470                    CLR     HIGOCT            ;CLEAR HIGH BINARY BITS
030566  022761  040515  006132            CMP     #"MA,PRMNEM(R1)   ;SEE IF PARAMETER IS (MA)
030574  001003                            BNE     1$                ;BP IF NOT (MA)
030576  016137  005714  005470            MOV     PRMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
030604  012746  005466            1$:     MOV     #LOWOCT,-(SP)     ;PUT ADDRESS OF BINARY VALUE ON STACK
030610  004737  052640                    JSR     PC,@#$DB2O        ;CONVERT BINARY TO OCTAL ASCII
030614  004737  053154                    JSR     PC,@#$SUPRS       ;TYPE YYYYYYY, SUPPRESSING LEADING O'S
030620  000207                            RTS     PC                ;RETURN


                                        ;;*****************************************************************
                                        .SBTTL  TYPPAT - TYPE CURRENT WORD OF DATA PATTERN 15
                                        ;*THIS SUBROUTINE TYPES THE LINE : WORD XX = YYYYYY,
                                        ;*WHERE XX IS THE NUMBER OF THE CURRENT WORD OF
                                        ;*USER-DEFINED PATTERN 15, AND YYYYYY IS ITS 16-BIT
                                        ;*VALUE, IN OCTAL DIGITS. <CR> AND <LF> ARE NOT TYPED.
                                        ;*ON ENTRY, R1 MUST CONTAIN THE INDEX OF THE CURRENT
                                        ;*WORD IN THE PATTERN 15 TABLE.
                                        ;*  CALL -      JSR       PC,TYPPAT
                                        ;;*****************************************************************

030622                                  TYPPAT:
030622  104401  012436                    TYPE    ,WORDSP           ;TYPE "WORD "
030626  010146                            MOV     R1,-(SP)          ;PUT INDEX ONTO STACK
030630  006216                            ASR     (SP)              ;DIVIDE BY TWO FOR WORD NO.
030632  104403                            TYPOS                     ;GO TYPE WORD NUMBER
030634     002                            .BYTE   2                 ;DIGIT COUNT = 2 FOR TYPOC
030635     001                            .BYTE   1                 ;TELL TYPOS TO TYPE LEADING ZEROS
030636  104401  012444                    TYPE    ,EQUALS           ;TYPE " = "
030642  016146  007136                    MOV     PAT15(R1),-(SP)   ;GET BINARY VALUE OF THIS WORD
030646  104403                            TYPOS                     ;TYPE VALUE IN OCTAL
030650     006                            .BYTE   6                 ;TYPE 6 DIGITS
030651     001                            .BYTE   1                 ;TYPE LEADING ZEROS
030652  000207                            RTS     PC                ;RETURN


                                        ;;*****************************************************************
                                        .SBTTL  MODP15 - MODIFY USER-DEFINED PATTERN 15
                                        ;*THIS SUBROUTINE DETERMINES WHETHER OR NOT USER-DEFINED
                                        ;*DATA PATTERN 15 SHOULD BE OPENED FOR MODIFICATION, AND
                                        ;*IF SO, IT REQUESTS THE NEW VALUES AND CONTROLS THEIR
                                        ;*INPUT ON THE TTY. IT CHECKS VALUES FOR VALIDITY, AND
                                        ;*LOADS THE 16 WORDS INTO THE PATTERN 15 TABLE (PAT15).
                                        ;*IF THE OPERATOR ANSWERS THE REQUEST FOR A PATTERN WORD
                                        ;*WITH EXCLAMATION POINT (!) (CR), THE SUBROUTINE
                                        ;*PROPAGATES THE LAST WORD TYPED INTO ALL REMAINING
                                        ;*WORDS OF THE PATTERN 15 TABLE.
                                        ;*  CALL -    JSR       PC,MODP15
```

```
                                        ;;*****************************************************************
                                        MODP15:
                                                MOV     R0,-(SP)        ;SAVE R0
                                                MOV     R1,-(SP)        ;SAVE R1
                                                MOV     R2,-(SP)        ;SAVE R2
                                        ;SEE IF PARAMETER IS PT
030662  022761  052120  006132                 CMP     #"PT,PRMNEM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
030670  001127                                BNE     22$             ;BR IF NOT (PT)
                                        ;SEE IF PATTERN 15 IS SPECIFIED
030672  032737  100000  005740                 BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
030700  001523                                BEQ     22$             ;BR IF NOT SPECIFIED
                                        ;SEE IF PATTERN 15 SHOULD BE MODIFIED
030702  104401  011126                 4$:    TYPE    ,MDFY15         ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
030706  004737  030174                        JSR     PC,RDCHRS       ;READ RESPONSE
030712  031160                                24$                     ;(↑C) RETURN ADDRESS
030714  031170                                26$                     ;(↑Z) RETURN ADDRESS
030716  030754                                8$                      ;(↑U) OR ERROR RETURN ADDRESS
030720  005700                                TST     R0              ;SEE IF NULL INPUT
030722  001512                                BEQ     22$             ;BR IF MODIFICATION NOT REQUESTED
030724  022737  000115  005262                 CMP     #'M,BUFFO       ;SEE IF (M) TYPED
030732  001405                                BEQ     6$              ;BR IF MODIFICATION REQUESTED
030734  104401  005262                        TYPE    ,BUFFO          ;ECHO BAD INPUT
030740  104401  001314                        TYPE    ,SQUES
030744  000756                                BR      4$              ;GO ASK AGAIN
                                        ;MODIFY PATTERN 15
030746  104401  011076                 6$:    TYPE    ,SELP15         ;TYPE "MODIFY PATTERN 15"
030752  005001                                CLR     R1              ;INITIALIZE WORD INDEX
030754  004737  030622                 8$:    JSR     PC,TYPPAT       ;TYPE CURRENT WORD AND VALUE
030760  104401  012430                        TYPE    ,SPACE1         ;TYPE A SPACE
030764  104401  012450                        TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
                                        ;READ AND CHECK INPUT, IF ANY
030770  004737  030174                        JSR     PC,RDCHRS       ;READ NEW DATA PATTERN WORD
030774  031160                                24$                     ;(↑C) RETURN ADDRESS FOR RDCHRS
030776  031170                                26$                     ;(↑Z) RETURN ADDRESS FOR RDCHRS
031000  030754                                8$                      ;(↑U) OR ERROR RETURN ADDR. FOR RDCHRS
031002  005700                                TST     R0              ;SEE IF ANY INPUT
031004  001006                                BNE     12$             ;BR IF ANY INPUT
031006  062701  000002                 10$:   ADD     #2,R1           ;INCREMENT WORD INDEX
031012  022701  000040                        CMP     #32.,R1         ;SEE IF ALL DONE
031016  001454                                BEQ     22$             ;BR IF DONE, TO RETURN
031020  000755                                BR      8$              ;CONTINUE WITH NEXT WORD
031022  022737  000041  005262         12$:   CMP     #'!,BUFFO       ;SEE IF (!) TYPED
031030  001431                                BEQ     16$             ;BR TO PROPAGATE CURRENT VALUE
031032  005002                                CLR     R2              ;INIT. (!) INDICATOR
031034  122760  000041  005261                CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
031042  001004                                BNE     14$             ;BR IF NOT (!)
031044  105060  005261                        CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
031050  005300                                DEC     R0              ;DECREMENT CHAR COUNT
031052  005202                                INC     R2              ;SET (!) INDICATOR
031054  022700  000006                 14$:   CMP     #6,R0           ;SEE HOW MANY CHARS NOW
031060  002426                                BLT     20$             ;BR IF TOO MANY
031062  012746  005262                        MOV     #BUFFO,-(SP)    ;GET BUF. ADDR. ON STACK FOR OCTBIN
031066  004737  051702                        JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
031072  031136                                20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
031074  012600                                MOV     (SP)+,R0        ;GET BINARY VALUE
```

```
 7259  031076  005737  052034          TST     $HIOCT          ;SEE IF VALUE EXCEEDS 16 BITS
 7260  031102  001015                  BNE     20$             ;BR TO ECHO BAD INPUT
 7261  031104  010561  007136          MOV     R0,PAT15(R1)    ;PUT NEW WORD VALJE INTO TABLE
 7262  031110  005702                  TST     R2              ;SEE IF (!) WAS TYPED
 7263  031112  001735                  BEQ     10$             ;BR IF (!) WAS NOT TYPED
 7264                          ;PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE
 7265  031114  022701  000036  16$:    CMP     #30.,R1         ;SEE IF ALL DONE YET
 7266  031120  001413                  BEQ     22$             ;BR IF DONE
 7267  031122  016161  007136  007140  MOV     PAT15(R1),PAT15+2(R1) ;PROPAGATE TO NEXT WORD
 7268  031130  062701  000002          ADD     #2,R1           ;INCREMENT WORD INDEX
 7269  031134  000767                  BR      16$             ;LOOP UNTIL DONE
 7270                          ;ECHO BAD INPUT
 7271  031136  104401  005262  20$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
 7272  031142  104401  001314          TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 7273  031146  000702                  BR      8$              ;BR TO ASK AGAIN
 7274                          ;NORMAL RETURN
 7275  031150  012602  22$:    MOV     (SP)+,R2        ;RESTORE R2
 7276  031152  012601          MOV     (SP)+,R1        ;RESTORE R1
 7277  031154  012600          MOV     (SP)+,R0        ;RESTORE R0
 7278  031156  000207          RTS     PC              ;RETURN
 7279                          ;(↑C) RETURN
 7280  031160  012766  013660  000006  24$:    MOV     #DRVTST,6(SP)   ;PC FOR (↑C) RETURN
 7281  031166  000770          BR      22$             ;BR TO RETURN
 7282                          ;(↑Z) RETURN
 7283  031170  012766  015304  000006  26$:    MOV     #ASKMDE,6(SP)   ;PC FOR (↑Z) RETURN
 7284  031176  000764          BR      22$             ;BR TO RETURN
 7285
 7286
 7287
 7288                          ;;*****************************************************************
 7289                          ;SBTTL  C-KPRM - CHECK VALUE OF INPUT PARAMETER
 7290                          ;*THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
 7291                          ;*HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
 7292                          ;*LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
 7293                          ;*INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
 7294                          ;*TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
 7295                          ;*IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
 7296                          ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
 7297                          ;*THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
 7298                          ;*VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
 7299                          ;* CALL -      JSR     PC,CHKPRM
 7300                          ;*            <ERROR RETURN ADDRESS>
 7301                          ;*            RETURN
 7302                          ;;*****************************************************************
 7303
 7304  031200  104407  CHKPRM: SAVREG                          ;SAVE R0-R5
 7305  031202  010102          MOV     R1,R2                   ;GET COPY OF INDEX
 7306  031204  006302          ASL     R2                      ;DOUBLE IT
 7307  031206  022761  040515  006132  CMP     #"MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
 7308  031214  001422          BEQ     20$                     ;BR IF (MA)
 7309  031216  005737  005470          TST     HIGOCT          ;SEE IF HIGH BITS = 0
 7310  031222  001403          BEQ     18$                     ;BR IF ZERO
 7311  031224  017616  000000  16$:    MOV     @(SP),(SP)      ;GET ERROR RETURN PC
 7312  031230  000475          BR      30$                     ;GO TO RETURN
 7313                          ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
 7314  031232  023762  005466  006022  18$:    CMP     LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
```

```
 7315  031240  103771                          BLO     16$             ;BR IF INPUT VALUE TOO SMALL
 7316  031242  023762  005466  006024          CMP     LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
 7317  031250  101365                          BHI     16$             ;BR IF INPUT VALUE IS TOO LARGE
 7318                                  ;UPDATE 16-BIT PARAMETER VALUE IN LIST
 7319  031252  013761  005466  005712          MOV     LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
 7320  031260  000457                          BR      28$             ;BR TO RETURN
 7321                                  ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
 7322  031262  032737  000001  005466  20$:    BIT     #BIT0,LOWOCT    ;SEE IF MA IS ODD
 7323  031270  001355                          BNE     16$             ;BR IF MA IS ODD
 7324  031272  023762  005470  006024          CMP     HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
 7325  031300  103751                          BLO     16$             ;BR IF HIGH WORD IS TOO SMALL
 7326  031302  001004                          BNE     24$             ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
 7327  031304  023762  005466  006022          CMP     LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
 7328  031312  103744                          BLO     16$             ;BR IF LOW WORD IS TOO SMALL
 7329  031314  023762  005470  006030  24$:    CMP     HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
 7330  031322  101340                          BHI     16$             ;BR IF HIGH WORD TOO BIG
 7331  031324  001004                          BNE     26$             ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
 7332  031326  023762  005466  006026          CMP     LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
 7333  031334  101333                          BHI     16$             ;BR IF LOW WORD IS TOO LARGE
 7334                                  ;UPDATE 32-BIT PARAMETER VALUE IN LIST
 7335  031336  013761  005466  005712  26$:    MOV     LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
 7336  031344  013761  005470  005714          MOV     HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
 7337                                  ;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
 7338  031352  004737  031430                  JSR     PC,MXWRDC       ;GET WORD COUNT IN R1-R0
 7339  031356  104401  010627                  TYPE    ,MAWRDC         ;TYPE "MAX WORD COUNT = "
 7340  031362  010037  003166                  MOV     R0,SUML01
 7341  031366  010137  003170                  MOV     R1,SUMHI1
 7342  031372  012746  003166                  MOV     #SUML01,-(SP)   ;PUT POINTER ON STACK
 7343  031376  004737  052640                  JSR     PC,$DB2O        ;CONVERT TO OCTAL
 7344  031402  004737  053154                  JSR     PC,$SUPRS       ;TYPE IT
 7345  031406  104401  001315                  TYPE    ,$CRLF          ;TYPE (CR),(LF)
 7346  031412  062766  000002  000014          ADD     #2,14(SP)       ;INCREMENT R1 INDEX
 7347  031420  062716  000002          28$:    ADD     #2,(SP) ;GET NORMAL RETURN PC
 7348  031424  104410                  30$:    RESREG                  ;RESTORE R0-R5
 7349  031426  000207                          RTS     PC              ;RETURN
 7350
 7351
 7352
 7353                                  ;;*****************************************************************
 7354                                  ;*MXWRDC - GET MAX WORD COUNT FOR CURRENT MEM LIMITS AND CURRENT MA,
 7355                                  ;*AND LEAVE THE WORD COUNT IN R1-R0. NO REGISTERS ARE SAVED.
 7356                                  ;;*****************************************************************
 7357  031430  013700  006106  MXWRDC: MOV     MAHILM,R0       ;GET LO BITS OF MEM LIM
 7358  031434  013701  006110          MOV     MAHILM+2,R1     ;HI BITS OF MEM LIM
 7359  031440  163700  005742          SUB     MA,R0           ;SUBTRACT MA FROM MAHILM
 7360  031444  005601                  SBC     R1
 7361  031446  163701  005744          SUB     MA+2,R1
 7362  031452  100413                  BMI     27$             ;IF NEGATIVE, RETURN
 7363  031454  000241                  CLC                     ;DIVIDE BY 2 TO GET WORDS
 7364  031456  006001                  ROR     R1
 7365  031460  006000                  ROR     R0
 7366  031462  062700  000001          ADD     #1,R0           ;INCREMENT BY 1 WORD
 7367  031466  005501                  ADC     R1
 7368  031470  005701                  TST     R1
 7369  031472  001403                  BEQ     27$             ;BR IF WORD COUNT < 65,536 DEC
 7370  031474  005000                  CLR     R0              ;MAKE WORD COUNT = 65,536
```

```
 7371  031476  012701  000001                    MOV     #1,R1
 7372  031502  000207                    27$:    RTS     PC              ;RETURN
 7373
 7374
 7375
 7376                                    ;;*****************************************************************
 7377                                    .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
 7378                                    ;*THIS SUBROUTINE TYPES "DRIVE SER. NO.  XXX" (IN DECIMAL), WITH LEADING
 7379                                    ;*ZEROS SUPPRESSED.
 7380                                    ;;*****************************************************************
 7381  031504  104407                    DRVSER: SAVREG                  ;SAVE R0-R5
 7382  031506  004737  027300                    JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
 7383  031512  112765  000141  000001            MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
 7384  031520  004737  037662                    JSR     PC,DRVCAL       ;READ STATUS OF THIS DRIVE
 7385  031524  104401  011345                    TYPE    ,DRIV           ;TYPE "DRIVE"
 7386  031530  104401  011363                    TYPE    ,SERNM          ;TYPE " SER. NO.  "
 7387  031534  016501  000054                    MOV     P.A11(R5),R1    ;GET "A" STATUS BYTE 11
 7388  031540  012704  052742                    MOV     #$OCTVL,R4      ;GET ADDR OF CHAR BUFFER
 7389  031544  010446                            MOV     R4,-(SP)        ;STORE IT ON STACK FOR $SUPRS
 7390  031546  012703  000003                    MOV     #3,R3           ;INIT CHAR COUNT
 7391  031552  006101                            ROL     R1              ;INITIALIZE BIT POSITIONS
 7392  031554  006101                            ROL     R1
 7393  031556  006101                    4$:     ROL     R1              ;GET NEXT 4 BITS
 7394  031560  006101                            ROL     R1
 7395  031562  006101                            ROL     R1
 7396  031564  006101                            ROL     R1
 7397  031566  010100                            MOV     R1,R0           ;GET A WORKING COPY
 7398  031570  042700  177760                    BIC     #177760,R0      ;CLEAR ALL BUT LOW 4 BITS
 7399  031574  052700  000060                    BIS     #'0,R0          ;CONVERT A DIGIT TO ASCII
 7400  031600  110024                            MOVB    R0,(R4)+        ;PUT ASCII DIGIT INTO CHAR BUFFER
 7401  031602  005303                            DEC     R3              ;DECREMENT CHAR COUNT
 7402  031604  001364                            BNE     4$              ;BR IF NOT 3 CHARS YET
 7403  031606  105014                            CLRB    (R4)            ;INSERT NULL TERMINATOR
 7404  031610  004737  053154                    JSR     PC,@#$SUPRS     ;TYPE DRIVE SER. NUMBER
 7405  031614  104401  001315                    TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7406  031620  104410                            RESREG                  ;RESTORE R0-R5
 7407  031622  000207                            RTS     PC              ;RETURN
 7408
 7409
 7410
 7411                                    ;;*****************************************************************
 7412                                    .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
 7413                                    ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXXX" (IN OCTAL),
 7414                                    ;*WITH LEADING ZEROS SUPPRESSED.
 7415                                    ;;*****************************************************************
 7416  031624  004737  027300            CRTSER: JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
 7417  031630  105065  000007                    CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
 7418  031634  105737  003115                    TSTB    FORMAT          ;CHECK THE ACTUAL FORMAT
 7419  031640  001402                            BEQ     4$              ;BR IF 22 SECTORS
 7420  031642  105265  000004                    INCB    P.SECT(R5)      ;IF 20 SECTORS, READ SECTOR 1
 7421  031646  112765  000121  000001    4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 7422  031654  012765  000632  000002            MOV     #LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)
 7423  031662  112765  000002  000005            MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
 7424  031670  012765  063526  000010            MOV     #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
 7425  031676  012765  177776  000012            MOV     #-2,P.WC(R5)    ;SET WORD COUNT TO READ 2 WORDS
 7426  031704  004737  037662                    JSR     PC,DRVCAL       ;READ SERIAL NO. IN BSF
```

```
 7427  031710  104401  011354                  TYPE    ,CART           ;TYPE   "CART."
 7428  031714  104401  011363                  TYPE    ,SERNM          ;TYPE "SER. NO.  "
 7429  031720  012746  063526                  MOV     #RWBUF,-(SP)    ;GET POINTER FOR $DB2O
 7430  031724  004737  052640                  JSR     PC,@#$DB2O      ;CONVERT BINARY TC OCTAL
 7431  031730  004737  053154                  JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
 7432  031734  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF.
 7433  031740  000207                          RTS     PC              ;RETURN
 7434
 7435
 7436
 7437                                  ;;*****************************************************************
 7438                                  .SBTTL  SEEKER - PERFORM IMPLICIT OR EXPLICIT SEEK
 7439                                  ;*THIS SUBROUTINE CHECKS SWR BIT 7, AND IF IT IS 0, AN IMPLICIT
 7440                                  ;*SEEK VIA THE READ HEADER COMMAND IS PERFORMED. IF THE HEADER
 7441                                  ;*DOES NOT CONTAIN THE CORRECT CYLINDER NO. AN ERROR IS REPORTED.
 7442                                  ;*IF SWR BIT 7 = 1, AN EXPLICIT SEEK COMMAND IS PERFORMED.
 7443                                  ;*ON ENTRY TO THE SUBROUTINE, THE DRIVE AND CYLINDER NUMBERS
 7444                                  ;*MUST BE PRE-LOADED INTO THE PARAMETER BLOCK.
 7445                                  ;* CALL -    JSR     PC,SEEKER
 7446                                  ;;*****************************************************************
 7447
 7448  031742  010046                  SEEKER: MOV     R0,-(SP)        ;SAVE R0
 7449  031744  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
 7450  031752  032777  000200  147160          BIT     #BIT07,@SWR     ;SEE IF EXPLICIT SEEKS REQUESTED
 7451  031760  001023                          BNE     2$              ;BR IF EXPLICIT SEEKS
 7452  031762  112765  000125  000001          MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
 7453  031770  004737  037662                  JSR     PC,DRVCAL       ;READ A HEADER
 7454  031774  016200  000024                  MOV     RKDB(R2),R0     ;GET WORD 1 OF ACTUAL HEADER
 7455  032000  020065  000002                  CMP     R0,P.CYLN(R5)   ;COMPARE TO EXPECTED CYLINDER
 7456  032004  001413                          BEQ     4$              ;BR IF EQUAL
 7457  032006  004737  041074                  JSR     PC,REPSUP       ;STORE PREV AND CURRENT CMNDS
 7458  032012  016537  000002  001174          MOV     P.CYLN(R5),$REG5 ;GET GOOD CYL NO.
 7459  032020  010037  001176                  MOV     R0,$REG6        ;GET BAD CYL NO.
 7460  032024  104062                          ERROR   62              ;CYLINDER MISCOMPARE
 7461  032026  000402                          BR      4$
 7462  032030  004737  037662          2$:     JSR     PC,DRVCAL       ;PERFORM COMMAND
 7463  032034  012600                  4$:     MOV     (SP)+,R0        ;RESTORE R0
 7464  032036  000207                          RTS     PC              ;RETURN
 7465
 7466
 7467
 7468                                  ;;*****************************************************************
 7469                                  .SBTTL  STALL - STALL FOR ST UNIT STALL TIMES
 7470                                  ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
 7471                                  ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
 7472                                  ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
 7473                                  ;* CALL -   JSR     PC,STALL
 7474                                  ;;*****************************************************************
 7475
 7476  032040  010046                  STALL:  MOV     R0,-(SP)        ;SAVE R0
 7477  032042  010146                          MOV     R1,-(SP)        ;SAVE R1
 7478  032044  032777  000400  147066          BIT     #BIT08,@SWR     ;APPLY RANDOM STALL ?
 7479  032052  001407                          BEQ     1$              ;BR IF NOT RANDOM
 7480  032054  004737  052036                  JSR     PC,$RAND        ;GENERATE PSEUDO-RANDOM NUMBER
 7481  032060  013700  052136                  MOV     $LONUM,R0       ;GET IT INTO R0
 7482  032064  006200                          ASR     R0              ;SCALE IT DOWN
```

```
7483  032066  006200                        ASR     R0
7484  032070  000403                        BR      2$              ;GO STALL WITH RANDOM NO.
7485  032072  013700  005502        1$:     MOV     STALLS,R0       ;GET REQUESTED NO. OF STALLS
7486  032076  001406                        BEQ     6$              ;RETURN IF NO STALL REQUIRED
7487  032100  012701  000016        2$:     MOV     #14.,R1         ;SET CONSTANT FOR 40 US
7488  032104  005301                4$:     DEC     R1              ;INNER LOOP COUNTER
7489  032106  001376                        BNE     4$              ;INNER LOOP BR
7490  032110  005300                        DEC     R0              ;OUTER LOOP COUNTER
7491  032112  001372                        BNE     2$              ;OUTER LOOP BR
7492  032114  012601                6$:     MOV     (SP)+,R1        ;RESTORE R1
7493  032116  012600                        MOV     (SP)+,R0        ;RESTORE R0
7494  032120  000207                        RTS     PC              ;RETURN
7495
7496
7497
7498                                ;;*****************************************************************
7499                                .SBTTL  RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
7500                                ;*THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
7501                                ;*ADDRESS, AND LEAVES IT IN "CYLNDR". IT REQUIRES THE SYSMAC
7502                                ;*SUBROUTINE $RAND.
7503                                ;* CALL -   JSR     PC,RNDADR
7504                                ;;*****************************************************************
7505  032122  004737  052036        RNDADR: JSR     PC,$RAND        ;GENERATE 2 16-BIT RANDOM NUMBERS
7506  032126  013737  052136  005504        MOV     $LONUM,CYLNDR   ;GET A RANDOM NUMBER
7507  032134  042737  177000  005504        BIC     #177000,CYLNDR  ;SCALE IT TO 9 BITS
7508  032142  022737  000632  005504        CMP     #632.CYLNDR     ;SEE IF CYL IS TOO BIG
7509  032150  002003                        BGE     2$              ;BR IF CYL IS OK
7510  032152  042737  000400  005504        BIC     #BIT08,CYLNDR   ;SCALE DOWN TO A VALID CYLINDER
7511  032160  000207                2$:     RTS     PC              ;RETURN
7512
7513
7514
7515                                .SBTTL  ROUTINES TO HANDLE KW11-L OR P CLOCK
7516
7517                                ;;*****************************************************************
7518                                ;*CLKON - ENABLE KW11-L OR P CLOCK INTERRUPT
7519                                ;;*****************************************************************
7520  032162  105737  003122        CLKON:  TSTB    PCLKF           ;SEE WHICH CLOCK WILL BE USED
7521  032166  001004                        BNE     1$              ;BR IF P-CLOCK TO BE USED
7522  032170  012777  000100  150742        MOV     #100,@LKS       ;L-CLOCK, ENABLE INTERRUPT
7523  032176  000207                        RTS     PC              ;RETURN
7524  032200  012777  174575  150736  1$:     MOV     #-1667.,@PKSB   ;SET 60 HZ. COUNT
7525  032206  005737  000166                TST     HZ              ;DETERMINE LINE FREQUENCY
7526  032212  001403                        BEQ     3$              ;BR IF 60 HZ.
7527  032214  012777  174060  150722        MOV     #-2000.,@PKSB   ;SET 50 HZ. COUNT
7528  032222  012777  000131  150712  3$:     MOV     #131,@PKS       ;ENABLE INTERRUPT,CNT UP,REP. INT.
7529                                                                ;  100 KHZ.,AND RUN
7530  032230  000207                        RTS     PC              ;RETURN
7531
7532                                ;;*****************************************************************
7533                                ;*CLOCK - HANDLE KW11-L OR P CLOCK INTERRUPT
7534                                ;;*****************************************************************
7535  032232  005200                CLOCK:  INC     R0              ;SIGNIFY A CLOCK TICK
7536  032234  000002                        RTI                     ;RETURN FROM INTR
7537
7538                                ;;*****************************************************************
```

```
                                ;*CLKOF - DISABLE KW11-L OR P CLOCK INTERRUPT
                                ;;*****************************************************************
032236  105737  003122  CLKOF:  TSTB    PCLKF           ;SEE WHICH CLOCK USED
032242  001003                  BNE     1$              ;BR IF P-CLOCK USED
032244  005077  150670          CLR     @LKS            ;DISABLE KW11-L INTR
032250  000207                  RTS     PC              ;RETURN
032252  005077  150664  1$:     CLR     @PKS            ;DISABLE KW11-P INTR
032256  000207                  RTS     PC              ;RETURN


                                ;;*****************************************************************
                                .SBTTL  CALBRT - CALIBRATE THE SOFTWARE TIMER
                                ;*THIS SUBROUTINE CALIBRATES THE SOFTWARE TIMER USED IN THE TIMING
                                ;*TESTS. IT CALLS SUBROUTINE "TIMER" TO MEASURE THE TIME BETWEEN
                                ;* 2 TICKS OF THE KW11-L OR P CLOCK. THIS INTERVAL EQUALS THE LINE
                                ;*PERIOD. IN ALL, 128 MEASUREMENTS ARE MADE AND AVERAGED, AND THE
                                ;*RESULTANT CALIBRATION CONSTANT IS LEFT IN TCONHI-TCONLO (IN
                                ;* NANO-SEC). IF THE SUBROUTINE HAPPENS TO TIME OUT WAITING FOR
                                ;*A TICK OF THE CLOCK, A RETURN IS MADE TO THE ADDRESS LISTED
                                ;*IMMED. AFTER THE CALL TO CALBRT.
                                ;* CALL -     JSR     PC,CALBRT
                                ;*             <ERROR RETURN ADDRESS>
                                ;;*****************************************************************

032260  104407          CALBRT: SAVREG                  ;SAVE R0-R5
032262  005002                  CLR     R2              ;INIT CALIBRATION LOOP CONSTANT
032264  005037  003166          CLR     SUMLO1          ;INIT SUM TO 0
032270  005037  003170          CLR     SUMHI1
032274  005001          2$:     CLR     R1              ;INIT TIME-OUT INDICATOR
032276  005000                  CLR     R0              ;INIT CLOCK INTR INDIC.
                                ;GET IN SYNCH WITH CLOCK
032300  004737  032162          JSR     PC,CLKON        ;ENABLE KW11-L OR P CLOCK INTR
032304  005700          4$:     TST     R0              ;SEE IF CLOCK INTR REC'D YET
032306  001013                  BNE     8$              ;BR IF INTR REC'D
032310  005201                  INC     R1              ;INCREMENT CLOCK TIME-OUT INDICATOR
032312  001374                  BNE     4$              ;BR IF NO TIME-OUT
032314  104401  011531  6$:     TYPE    ,CLKFAL         ;TIMED-OUT ON KW11-L OR P INTERRUPT
032320  104401  011466          TYPE    ,TIMSKP         ;SAY ALL TIMING TESTS WILL BE SKIPPED
032324  105037  003123          CLRB    DOTIM           ;DON'T ALLOW TIMING TESTS
032330  017616  000000          MOV     @(SP),(SP)      ;FIX UP ERROR RETURN PC
032334  000464                  BR      18$             ;GO TO RETURN
                                ;RUN TIMER TO NEXT CLOCK TICK
032336  005000          8$:     CLR     R0              ;INIT CLOCK INTR INDIC
032340  005004                  CLR     R4              ;INIT HI BITS OF CYCLE COUNT
032342  005005                  CLR     R5              ;LO BITS
032344  004737  032516          JSR     PC,TIMER        ;COUNT CYCLES TIL NEXT CLOCK INTR
032350  032314                  6$                      ;ERROR RETURN ADDRESS
                                ;ADD MEASURED CYCLES TO SUM
032352  060537  003166          ADD     R5,SUMLO1       ;ADD MEASUREMENT TO SUM
032356  005537  003170          ADC     SUMHI1
032362  060437  003170          ADD     R4,SUMHI1
032366  103752                  BCS     6$              ;BR IF SUM OVERFLOW
032370  005202                  INC     R2              ;INCR CALIBRATION LOOP COUNT
032372  022702  000200          CMP     #128.,R2        ;SEE IF 128 MEASUREMENTS MADE YET
032376  001336                  BNE     2$              ;BR IF NOT YET
```

```
                                        ;TAKE AVERAGE OF 128 MEASUREMENTS
032400  012702  000007                  MOV     #7,R2           ;SET DIVIDE LOOP COUNT
032404  000241                  12$:    CLC
032406  006037  003170                  ROR     SUMHI           ;DIVIDE SUM BY 2
032412  006037  003166                  ROR     SUMLOI
032416  005302                          DEC     R2              ;DECREMENTT LOOP COUNT
032420  001371                          BNE     12$             ;BR IF DIVISION NOT DONE YET
                                        ;DIVIDE LINE PERIOD IN N-SEC BY TIMER CYCLE COUNT TO GET CALIB. CONST.
032422  005000                          CLR     R0              ;SET DIVIDEND = 16666667 NS
032424  005001                          CLR     R1
032426  012702  000376                  MOV     #376,R2
032432  012703  050053                  M.      #↑D20523,R3
032436  005737  000166                  TST     HZ              ;DETERMINE THE LINE FREQUENCY
032442  001404                          BEQ     16$             ;BR IF 60 HZ.
032444  012702  000461                  MOV     #461,R2         ;SET DIVIDEND = 20000000 NS
032450  012703  026400                  MOV     #↑D11520,R3
032454  013704  003170          16$:    MOV     SUMHI1,R4       ;SET DIVISOR
032460  013705  003166                  MOV     SUMLOI,R5
032464  004737  052462                  JSR     PC,M.DPID       ;PERFORM THE DIVISION
032470  010237  003156                  MOV     R2,TCONHI       ;GET FINAL TIMING CONSTANT
032474  001307                          BNE     6$              ;BR IF HI BITS NOT 0
032476  010337  003154                  MOV     R3,TCONLO       ;GET LO BITS
032502  062716  000002                  ADD     #2,(SP)         ;FIX UP ERROR-FREE RETURN PC
032506  004737  032236          18$:    JSR     PC,CLKOF        ;DISABLE KW11-L OR P CLOCK INTERRUPT
032512  104410                          RESREG                  ;RESTORE R0-R5
032514  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        .SBTTL  TIMER - SOFTWARE TIMER SUBROUTINE
                                        ;*THIS SUBROUTINE LOOPS THROUGH THE SOFTWARE TIMING LOOP,
                                        ;*INCREMENTING THE CYCLE COUNT IN R4-R5 EACH LOOP, AND CHECKS
                                        ;*THE INTERRUPT INDICATION IN R0, FOR COMPLETION. IF A COUNT
                                        ;*OF 4,194,304 IS REACHED IN R4-R5, A TIME-OUT ERROR RETURN
                                        ;*IS MADE TO THE ADDRESS LISTED AFTER THE CALL .
                                        ;* CALL -    JSR    PC,TIMER
                                        ;*          <ERROR RETURN ADDRESS>
                                        ;;*****************************************************************
                                        ;*** SPECIAL TIMING LOOP ***
032516  062705  000001          TIMER:  ADD     #1,R5           ;INCREMENT LO CYCLE COUNT
032522  005504                          ADC     R4              ;ADD CARRY TO HI CYCLE COUNT
032524  032704  000100                  BIT     #BIT06,R4       ;SEE IF TIMED-OUT WAITING FOR INTR
032530  001005                          BNE     4$              ;BR IF TIME-OUT
032532  005700                          TST     R0              ;SEE IF INTERRUPT RECEIVED
032534  001770                          BEQ     TIMER           ;BR IF INTERRUPT NOT REC'D YET
                                        ;***************************
032536  062716  000002                  ADD     #2,(SP)         ;FIX UP ERROR-FREE RETURN PC
032542  000207                          RTS     PC              ;ERROP-FREE RETURN
032544  017616  000000          4$:     MOV     @(SP),(SP)      ;FIX UP ERROR RETURN PC
032550  000207                          RTS     PC              ;ERROR RETURN


                                        ;;*****************************************************************
                                        ;* INIVRB - INITIALIZE VARIABLES USED IN TIMING TESTS
```

```
                                ;;*****************************************************************
                        INITRB: CLR     BLWMIN          ;COUNT OF TIMES BELOW SPEC'D MIN
                                CLR     ABVMX1          ;COUNT OF FORWARD TIMES ABOVE SPEC'D MAX
                                CLR     ABVMX2          ;COUNT OF REVERSE TIMES ABOVE SPEC'D MAX
                                CLR     SUMLO1          ;FORWARD TIME SUM AND AVERAGE
                                CLR     SUMHI1
                                CLR     SUMLO2          ;REVERSE TIME SUM AND AVERAGE
                                CLR     SUMHI2
                                MOV     #-1,MINIL1      ;MIN MEAS'D FORWARD TIME
                                MOV     #-1,MINIH1
                                CLR     MAXIL1          ;MAX MEAS'D FORWARD TIME
                                CLR     MAXIH1
                                MOV     #-1,MINIL2      ;MIN MEAS'D REVERSE TIME
                                MOV     #-1,MINIH2
                                CLR     MAXIL2          ;MAX MEAS'D REVERSE TIME
                                CLR     MAXIH2
                                RTS     PC              ;RETURN



                                ;;*****************************************************************
                                .SBTTL  TIMSEK - MEASURE AN EXPLICIT SEEK TIME
                                ;*THIS SUBROUTINE MEASURES THE AMOUNT OF TIME REQUIRED TO PERFORM
                                ;*A SEEK TO THE CYLINDER ADDRESS IN P.CYLN IN THE PARAM BLK. THIS
                                ;*TIME IS RETURNED IN R0-R1, IN US. IF THE SEEK TIMES-OUT, OR IF
                                ;*THE TIME CALCULATION OVERFLOWS, A MESSAGE IS TYPED, AND AN ERROR
                                ;*RETURN IS MADE TO THE ADDRESS LISTED AFTER THE CALL.
                                ;* CALL -   JSR     PC,TIMSEK
                                ;*          <ERROR RETURN ADDRESS>
                                ;;*****************************************************************
                        TIMSEK: MOV     R2,-(SP)        ;SAVE R2
                                MOV     R5,-(SP)        ;SAVE R5
                                CLR     R0              ;INIT DRIVE INTR INDICATOR
                                CLR     R4              ;INIT HI CYCLE COUNT
033040  150132                  MOV     #SEEKHD,@RKVEC  ;SET SPECIAL RK06 SEEK TIMER HANDLER VECTOR
000117  000001                  MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
032726                          MOV     R5,2$           ;SET PARAM BLK ADDRESS FOR DRIVER
                                CLR     R5              ;INIT LO CYCLE COUNT
000000                          ADD     #RKCS1,R2       ;GET ADR OF RKCS1 IN R2 FOR SEEKHD
037776                          JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
050006                          JSR     PC,C.INIT       ;START THE SEEK
000000                  2$:     .WORD   0               ;PARAM BLK ADR GOES HERE
032516                          JSR     PC,TIMER        ;MEASURE TIME TIL DRIVE ATT'N
033004                          6$                      ;ERROR RETURN ADDRESS FOR TIMER
033504                          JSR     PC,CNVTIM       ;CONVERT MEAS'D CYCLES TO TIME
033004                          6$                      ;ERROR RETURN ADR FOR CNVTIM
010200                          MOV     R2,R0           ;PUT HI TIME BITS IN R0
010301                          MOV     R3,R1           ;PUT LO TIME BITS IN R1
000002  000004                  ADD     #2,4(SP)        ;FIX UP ERROR-FREE RETURN PC
044506  150044          4$:     MOV     #I.INTR,@RKVEC  ;RESTORE RK06 INTR HANDLER
                                MOV     (SP)+,R5        ;RESTORE R5
                                MOV     (SP)+,R2        ;RESTORE R2
000177  000001                  MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
037662                          JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
000207                          RTS     PC              ;RETURN
003026                  6$:     MOV     RKBAS,R2        ;GET RK06 REG ADDR
```

```
033010  042762  000100  000000          BIC     #IE.RKCS1(R2)   ;DISABLE RK06 INTERRUPT
033016  004737  041074                  JSR     PC,REPSJP       ;GET PREV AND CURRENT CMNDS
033022  004737  043600                  JSR     PC,TOPROC       ;GATHER STATUS
033026  104106                          ERROR   106             ;TIMED-OUT ON SEEK
033030  017666  000004  000004          MOV     @4(SP),4(SP)    ;FIX ERROR RETURN PC
033036  000747                          BR      4$              ;BR TO EXIT


                                ;;*****************************************************************
                                ;*SPECIAL INTERRUPT HANDLER FOR SEEKS IN SEEK TIMING TESTS
                                ;;*****************************************************************
033040  011203                  SEEKHD: MOV     (R2),R3         ;GET BITS OF RKCS1
033042  100002                          BPL     2$              ;BR IF CERR NOT SET
033044  000137  044506                  JMP     I.INTR          ;LET DRIVER PROCESS THE ERROR
033050  032703  040000          2$:     BIT     #DI,R3          ;SEE IF DRIVE INTR SET YET
033054  001401                          BEQ     4$              ;BR IF NOT YET
033056  005200                          INC     R0              ;SET RK06 INTR INDICATOR
033060  000002                  4$:     RTI                     ;RETURN FROM INTERRUPT


                                ;;*****************************************************************
                                ;*FDTERM - COMPUTE A TERM OF THE FORWARD SEEK AVERAGE FORMULA,
                                ;*AND ADD IT TO THE FORWARD SUM IN SUML01-SUMHI1-SUML02-SUMHI2.
                                ;;*****************************************************************
033062  104407                  FDTERM: SAVREG                  ;SAVE R0-R5
033064  010002                          MOV     R0,R2           ;GET MEASUREMENT INTO  R2-R3
033066  010103                          MOV     R1,R3
033070  005004                          CLR     R4
033072  013705  005532                  MOV     SCRACH,R5       ;GET COEFFICIENT
033076  006305                          ASL     R5              ;MULTIPLY BY 2
033100  004737  052422                  JSR     PC,M.DPIM       ;MULTIPLY COEFFICIENT
033104  060337  003174                  ADD     R3,SUMHI2       ;ADD TO FORWARD SUM
033110  005502                          ADC     R2
033112  060237  003172                  ADD     R2,SUML02
033116  005537  003170                  ADC     SUMHI1
033122  104410                          RESREG                  ;RESTORE R0-R5
033124  000207                          RTS     PC


                                ;;*****************************************************************
                                ;*RVTERM - COMPUTE  A TERM  OF THE REVERSE AVERAGE FORMULA,
                                ;*AND ADD IT TO THE REVERSE SUM IN MAXIL1-MAXIH1-MAXIL2-MAXIH2.
                                ;;*****************************************************************
033126  104407                  RVTERM: SAVREG           ;SAVE R0-R5
033130  010002                          MOV     R0,R2           ;GET MEASUREMENT INTO R2-R3
033132  010103                          MOV     R1,R3
033134  005004                          CLR     R4
033136  013705  005532                  MOV     SCRACH,R5       ;GET COEFFICIENT
033142  006305                          ASL     R5              ;MULTIPLY BY 2
033144  004737  052422                  JSR     PC,M.DPIM       ;MULTIPLY COEFFICIENT
033150  060337  003220                  ADD     R3,MAXIH2       ;ADD TO REVERSE SUM
033154  005502                          ADC     R2
033156  060237  003216                  ADD     R2,MAXIL2
033162  005537  003210                  ADC     MAXIH1
033166  104410                          RESREG                  ;RESTORE R0-R5
033170  000207                          RTS     PC
```

```
                                ;;*****************************************************************
                                ;* CMPTIM - COMPARE MEAS'D TIME TO MEAS'D MIN AND MAX
                                ;*AND REPLACE, IF NECESSARY. TIME IS IN R0-R1, POINTER
                                ;*TO LO MIN IS IN R3.
                                ;;*****************************************************************
033172  020063  000002  CMPTIM: CMP     R0,2(R3)        ;COMPARE HI BITS TO HI MIN
033176  101006                  BHI     6$              ;BR IF > MIN
033200  103402                  BLO     4$              ;BR IF < MIN
033202  020113                  CMP     R1,(R3)         ;COMPARE LO BITS TO LO MIN
033204  103003                  BHIS    6$              ;BR IF > OR = MIN
033206  010063  000002  4$:     MOV     R0,2(R3)        ;SET NEW MIN HI BITS
033212  010113                  MOV     R1,(R3)         ;SET NEW MIN LO BITS
033214  020063  000006  6$:     CMP     R0,6(R3)        ;COMPARE HI BITS TO HI MAX
033220  103410                  BLO     10$             ;BR IF < MAX
033222  101003                  BHI     8$              ;BR IF > MAX
033224  020163  000004          CMP     R1,4(R3)        ;COMPARE LO BITS TO LO MAX
033230  101404                  BLOS    10$             ;BR IF < OR = MAX
033232  010063  000006  8$:     MOV     R0,6(R3)        ;SET NEW MAX HI BITS
033236  010163  000004          MOV     R1,4(R3)        ;SET NEW MAX LO BITS
033242  000207          10$:    RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*GETAVG - COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES BY
                                ;*DIVIDING TIME SUMS BY NO. OF MEASUREMENTS (PASSED IN R3 ON ENTRY).
                                ;;*****************************************************************
033244  104407          GETAVG: SAVREG                  ;SAVE R0-R5
033246  010305                  MOV     R3,R5           ;LO DIVISOR = MEASUREMENT COUNT
033250  005000                  CLR     R0              ;SET DIVIDEND = FORWARD TIME SUM
033252  005001                  CLR     R1
033254  013702  003170          MOV     SUMHI1,R2
033260  013703  003166          MOV     SUMLO1,R3
033264  005004                  CLR     R4              ;HI DIVISOR = 0
033266  004737  052462          JSR     PC,M.DPID       ;PERFORM DIVISION
033272  010237  003170          MOV     R2,SUMHI1       ;STORE FORWARD TIME AVG
033276  010337  003166          MOV     R3,SUMLO1
033302  005000                  CLR     R0              ;SET DIVIDEND = REVERSE TIME SUM
033304  005001                  CLR     R1
033306  013702  003174          MOV     SUMHI2,R2
033312  013703  003172          MOV     SUMLO2,R3
033316  004737  052462          JSR     PC,M.DPID       ;PERFORM DIVISION
033322  010237  003174          MOV     R2,SUMHI2       ;STORE REVERSE TIME AVG
033326  010337  003172          MOV     R3,SUMLO2
033332  104410                  RESREG                  ;RESTORE R0-R5
033334  000207                  RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;SBTTL  TYPTMS - TYPE RESULTS OF SEEK TIMING MEASUREMENTS
                                ;*THIS SUBROUTINE TYPES THE MIN,MAX, AND AVG SEEK TIMES
```

```
                                ;*(IN DECIMAL) MEASURED IN ONE OF THE SEEK TIMING TESTS,
                                ;*IN BOTH THE FORWARD AND REVERSE DIRECTIONS. ALSO TYPED ARE
                                ;*THE MAX SPEC'D TIME AND THE NO. OF SEEKS EXCEEDING IT,
                                ;* AND THE TOTAL NO. OF MEASUREMENTS. POINTER TO SPECIFIED
                                ;*MAX MESSAGE MUST BE IN R3 ON ENTRY.
                                ;;***************************************************************

        033236                  TYPTMS:
                                ;TYPE FORWARD MEASUREMENTS
        033236  104401  011714          TYPE    .FORWRD         ;TYPE "***FORWARD DIRECTION***"
        033242  012701  003202          MOV     #MINIL1,R1      ;POINTER TO LO MIN
        033246  004737  033556          JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
        033252  104401  001315          TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
        033256  012701  003206          MOV     #MAXIL1,R1      ;POINTER TO LO MAX
        033262  004737  033602          JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
        033266  013746  003162          MOV     ABVMX1,-(SP)    ;GET COUNT OF TIMES ABOVE MAX
        033272  104405                  TYPDS                   ;CONVERT AND TYPE IT
        033274  010337  033402          MOV     R3,4$           ;SET POINTER TO SPEC'D MAX MESSAGE
        033300  104401                  TYPE                    ;TYPE " OF XXX ABOVE MAX OF XXXXX US "
        033402  000000          4$:     .WORD   0               ;SPEC'D MAX MSG POINTER GOES HERE
        033404  012701  003166          MOV     #SUMLO1,R1      ;SET POINTER TO LO AVG
        033410  004737  033626          JSR     PC,TYPAVG       ;TYPE AVG TIME
        033414  104401  001315          TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
                                ;TYPE REVERSE MEASUREMENTS
        033420  104401  011744          TYPE    ,REVRSE         ;TYPE "***REVERSE DIRECTION***"
        033424  012701  003212          MOV     #MINIL2,R1      ;POINTER TO LO MIN
        033430  004737  033556          JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
        033434  104401  001315          TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
        033440  012701  003216          MOV     #MAXIL2,R1      ;POINTER TO LO MAX
        033444  004737  033602          JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
        033450  013746  003164          MOV     ABVMX2,-(SP)    ;GET COUNT OF TIMES ABOVE MAX
        033454  104405                  TYPDS                   ;CONVERT AND TYPE IT
        033456  010337  033464          MOV     R3,6$           ;GET POINTER TO SPEC'D MAX MESSAGE
        033462  104401                  TYPE                    ;TYPE " OF XXX ABOVE MAX OF XXXXX US "
        033464  000000          6$:     .WORD   0               ;SPEC'D MAX MSG POINTER GOES HERE
        033466  012701  003172          MOV     #SUMLO2,R1      ;SET POINTER TO LO AVG
        033472  004737  033626          JSR     PC,TYPAVG       ;TYPE AVG TIME
        033476  104401  001315          TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
        033502  000207                  RTS     PC              ;RETURN



                                ;;***************************************************************
                                ;* CNVTIM - CONVERT SOFTWARE TIMER CYCLES TO TIME
                                ;*THIS SUBROUTINE MULTIPLIES THE CYCLE COUNT IN R4-R5 BY THE
                                ;*TIME CONSTANT IN TCONHI-TCONLO, TO OBTAIN TIME IN N-SEC. THIS
                                ;*IS THEN DIVIDED BY 1000 (DEC) TO OBTAIN TIME IN MICRO-SEC.
                                ;*THIS TIME IS RETURNED IN R2-R3. IF THE MULTIPLICATION OVERFLOWS
                                ;*32 BITS, AN ERROR RETURN IS MADE TO THE ADDRESS LISTED AFTER THE
                                ;*CALL TO CNVTIM.
                                ;*CALL -        JSR     PC,CNVTIM
                                ;*              <ERROR RETURN ADDRESS>
                                ;;***************************************************************
        033504  013702  003156  CNVTIM: MOV     TCONHI,R2       ;SET MULTIPLIER = TIME CONST
        033510  013703  003154          MOV     TCONLO,R3
        033514  004737  052422          JSR     PC,M.DPIM       ;MULTIPLY
```

```
                033520  005700                  TST     R0              ;MAKE SURE HI 2 WORDS ARE 0
                033522  001002                  BNE     4$              ;BR IF NOT 0
                033524  005701                  TST     R1
                033526  001403                  BEQ     6$
                033530  017616  000000  4$:     MOV     @(SP),(SP)      ;SET ERROR RETURN PC
                033534  000207                  RTS     PC              ;ERROR RETURN
                033536  005004  6$:     CLR     R4              ;SET HI BITS = 0
                033540  012705  001750          MOV     #1000.,R5       ;SET LO DIVISOR = 1000 (DEC)
                033544  004737  052462          JSR     PC,M.DPID       ;CONVERT TIME TO US
                033550  062716  000002          ADD     #2.,(SP)        ;FIX ERROR-FREE RETURN PC
                033554  000207                  RTS     PC              ;ERROR-FREE RETURN


                                        ;;*****************************************************************
                                        ;*TYPMIN - TYPE MIN MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
                                        ;*ON ENTRY.
                                        ;;*****************************************************************
                033556  104401  011774  TYPMIN: TYPE    ,MINEQ          ;TYPE "MIN = "
                033562  010146          MOV     R1,-(SP)        ;GET POINTER TO LO TIME
                033564  004737  052760  JSR     PC,@#$DB2D      ;CONVERT TO DEC
                033570  004737  053154  JSR     PC,@#$SUPRS     ;TYPE IT
                033574  104401  012021  TYPE    ,MICROS         ;TYPE "US "
                033600  000207          RTS     PC


                                        ;;*****************************************************************
                                        ;*TYPMAX - TYPE MAX MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
                                        ;*ON ENTRY.
                                        ;;*****************************************************************
                033602  104401  012003  TYPMAX: TYPE    ,MAXEQ          ;TYPE "MAX = "
                033606  010146          MOV     R1,-(SP)        ;GET POINTER TO LO TIME
                033610  004737  052760  JSR     PC,@#$DB2D      ;CONVERT TO DEC
                033614  004737  053154  JSR     PC,@#$SUPRS     ;TYPE IT
                033620  104401  012021  TYPE    ,MICROS         ;TYPE "US "
                033624  000207          RTS     PC


                                        ;;*****************************************************************
                                        ;*TYPAVG - TYPE AVERAGE MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
                                        ;*ON ENTRY.
                                        ;;*****************************************************************
                033626  104401  012012  TYPAVG: TYPE    ,AVGEQ          ;TYPE "AVG = "
                033632  010146          MOV     R1,-(SP)        ;GET POINTER TO LO TIME
                033634  004737  052760  JSR     PC,@#$DB2D      ;CONVERT TO DEC
                033640  004737  053154  JSR     PC,@#$SUPRS     ;TYPE IT
                033644  104401  012021  TYPE    ,MICROS         ;TYPE "US "
                033650  000207          RTS     PC


                                        ;;*****************************************************************
                                        ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF R0 INTO ALL
```

```
                                        ;*256.(DEC) WORDS OF THE DATA BUFFER (RWBUF).
                                        ;:*****************************************************************
033652  104407                          LODSEC: SAVREG                  ;SAVE R0-R5
033654  012701  063526                          MOV     #RWBUF,R1       ;GET ADDRESS OF DATA BUF INTO R1
033660  010021                          2$:     MOV     R0,(R1)+        ;PUT WORD INTO BUFFER
033662  020127  064526                          CMP     R1,#RWBUF+512.  ;SEE IF 256 WORDS WRITTEN YET
033666  001374                                  BNE     2$              ;BR IF NOT DONE YET
033670  104410                                  RESREG                  ;RESTORE R0-R5
033672  000207                                  RTS     PC              ;RETURN


                                        ;:*****************************************************************
                                        ;* TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
                                        ;*FC, FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
                                        ;*TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
                                        ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.
                                        ;:*****************************************************************
033674  104407                          TRKCHK: SAVREG                  ;SAVE R0-R5
033676  016546  000004                          MOV     P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
033702  105065  000005                          CLRB    P.TRCK(R5)      ;CLEAR THE TRACK NO.
                                        ;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
033706  116500  000005                  2$:     MOVB    P.TRCK(R5),R0   ;GET TRACK NO. INTO R0
033712  062700  000100                          ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
033716  004737  033652                          JSR     PC,LODSEC       ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
033722  112765  000131  000001                  MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
033730  004737  037662                          JSR     PC,DRVCAL       ;PERFORM THE WRITE CHECK
033734  105265  000005                          INCB    P.TRCK(R5)      ;INCREMENT THE TRACK NO.
033740  122765  000003  000005                  CMPB    #3,P.TRCK(R5)   ;SEE IF DONE WITH ALL TRACKS
033746  001357                                  BNE     2$              ;BR IF NOT DONE YET
033750  012665  000004                          MOV     (SP)+,P.SECT(R5) ;RESTORE TRACK AND SECTOR PARAMETERS
033754  104410                                  RESREG                  ;RESTORE R0-R5
033756  000207                                  RTS     PC              ;RETURN


                                        ;:*****************************************************************
                                        ;*LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
                                        ;*INTO THE PATTERN 14 TABLE.
                                        ;:*****************************************************************
033760  104407                          LODP14: SAVREG                  ;SAVE R0-R5
033762  012701  000010                          MOV     #8.,R1          ;INIT LOOP COUNTER TO 8.
033766  012700  007076                          MOV     #PAT14,R0       ;GET ADDRESS OF PATTERN 14 BUFFER
033772  004737  052036                  4$:     JSR     PC,$RAND        ;GENERATE 2 16-BIT RANDOM NUMBERS
033776  013720  052136                          MOV     $LONUM,(R0)+    ;PUT ONE NUMBER INTO PATTERN
034002  013720  052134                          MOV     $HINUM,(R0)+    ;PUT OTHER NO. INTO PATTERN
034006  005301                                  DEC     R1              ;SEE IF 16 WORDS LOADED YET
034010  001370                                  BNE     4$              ;BR IF NOT YET
034012  104410                                  RESREG                  ;RESTORE R0-R5
034014  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.
                                        ;;*****************************************************************
034016  104407                  FINADR: SAVREG                  ;SAVE R0-R5
034020  013746  005570                  MOV     LASTWC,-(SP)    ;STORE WORD COUNT
034024  005416                          NEG     (SP)            ;MAKE IT POSITIVE
034026  005046                  18$:    CLR     -(SP)           ;MAKE ROOM ON STACK
034030  116616  000003                  MOVB    3(SP),(SP)      ;STORE NO. OF SECTORS TRANSFERRED
034034  005066  000002                  CLR     2(SP)           ;CLEAR LOCATION ON STACK
034040  116566  000004  000002          MOVB    P.SECT(R5),2(SP) ;STORE STARTING SECTOR
034046  066616  000002                  ADD     2(SP),(SP)      ;DETERMINE FINAL SECTOR ADDRESS
034052  005066  000002                  CLR     2(SP)           ;CLEAR NO. OF TRACKS TRANSFERRED
034056  012700  000026                  MOV     #22.,R0         ;SET FOR 22 SECTORS
034062  105737  003115                  TSTB    FORMAT          ;DETERMINE THE FORMAT
034066  001402                          BEQ     19$             ;BR IF 22 SECTORS
034070  012700  000024                  MOV     #20.,R0         ;SET FOR 20 SECTORS
034074  020016                  19$:    CMP     R0,(SP)         ;CHECK FOR SECTOR OVERFLOW
034076  101004                          BHI     20$             ;NO, CHECK IF SECTOR CORRECT
034100  160016                          SUB     R0,(SP)         ;DECREMENT SECTOR COUNT BY 20 OR 22
034102  005266  000002                  INC     2(SP)           ;INCREMENT TRACKS TRANSFERRED
034106  000772                          BR      19$             ;CHECK FOR SECTOR OVERFLOW
034110  112637  005567          20$:    MOVB    (SP)+,FINSEC    ;STORE FINAL SECTOR
034114  005046                          CLR     -(SP)           ;MAKE ROOM FOR TRACKS TRANSFERRED
034116  116513  000005                  MOVB    P.TRCK(R5),(SP) ;STORE STARTING TRACK
034122  066616  000002                  ADD     2(SP),(SP)      ;DETERMINE FINAL TRACK ADDRESS
034126  005066  000002                  CLR     2(SP)           ;CLEAR FINAL CYLINDER
034132  122716  000003          21$:    CMPB    #3,(SP)         ;CHECK FOR TRACK OVERFLOW
034136  101005                          BHI     22$             ;NO, CHECK FINAL TRACK
034140  162716  000003                  SUB     #3,(SP)         ;DECREMENT TRACK COUNT BY 3
034144  005266  000002                  INC     2(SP)           ;INCR CYL COUNT
034150  000770                          BR      21$             ;CHECK FOR TRACK OVERFLOW
034152  112637  005566          22$:    MOVB    (SP)+,FINTRK    ;STORE FINAL TRACK
034156  066516  000002                  ADD     P.CYLN(R5),(SP) ;CALCULATE FINAL CYLINDER
034162  011637  005564                  MOV     (SP),FINCYL     ;STORE FINAL  CYLINDER
034166  005726                          TST     (SP)+           ;CLEAN OFF STACK
034170  104410                          RESREG                  ;RESTORE R0-R5
034172  000207                          RTS     PC              ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  LODBUF - LOAD THE READ/WRITE DATA BUFFER
                                        ;*THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
                                        ;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
                                        ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
                                        ;*OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
                                        ;*IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
                                        ;*OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
                                        ;*ENTRY, ARE LOADED INTO THE BUFFER.
                                        ;;*****************************************************************
034174  104407                  LODBUF: SAVREG                  ;SAVE R0-R5
```

```
 8043  034176  013702  005562                    MOV     WDSXFR,R2       ;GET NO. OF WORDS
 8044  034202  005737  005534                    TST     PATRN           ;SEE IF QUICK VERIFY DATA TEST DESIRED
 8045  034206  001007                            BNE     3$              ;BR IF NOT QUICK VERIFY
 8046  034210  012700  006176                    MOV     #PAT00,R0       ;SET DATA PATTERN STARTING ADDRESS
 8047  034214  012746  000400                    MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
 8048  034220  012701  063526                    MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
 8049  034224  000463                            BR      30$             ;PROCEED
 8050  034226  005000                    3$:     CLR     R0              ;INIT PATTERN NUMBER
 8051  034230  032701  000001            4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
 8052  034234  001003                            BNE     6$              ;BR IF THIS BIT IS SET
 8053  034236  005200                            INC     R0              ;INCREMENT PATTERN NO.
 8054  034240  006201                            ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
 8055  034242  000772                            BR      4$              ;BR TO CHECK NEXT BIT
 8056  034244  006300                    6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
 8057  034246  006300                            ASL     R0
 8058  034250  006300                            ASL     R0
 8059  034252  006300                            ASL     R0
 8060  034254  006300                            ASL     R0
 8061  034256  062700  006176                    ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
 8062  034262  012746  000020                    MOV     #16.,-(SP)      ;SET PATTERN WORD COUNT
 8063  034266  013701  005572                    MOV     PMA,R1          ;SET BUFFER ADDRESS
 8064  034272  005737  055134                    TST     $KTI1           ;SEE IF MEM MGT PRESENT
 8065  034276  100036                            BPL     30$             ;BR IF NOT PRESENT
 8066                                    ;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
 8067  034300  013737  003176  172354            MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
 8068  034306  052737  000001  177572            BIS     #BIT0,@#SR0     ;TURN ON MEMORY MANAGEMENT
 8069  034314  042701  160000                    BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
 8070  034320  052701  140000                    BIS     #140000,R1
 8071  034324  010003                    22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8072  034326  011604                            MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8073  034330  012321                    24$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8074  034332  032701  020000                    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
 8075  034336  001405                            BEQ     26$             ;BR IF NO OVERFLOW
 8076  034340  062737  000200  172354            ADD     #200,@#KIPAR6   ;INCREMENT PAR BY 4K FOR NEW PAGE
 8077  034346  042701  020000                    BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
 8078  034352  005302                    26$:    DEC     R2              ;DECREMENT WORD COUNTER
 8079  034354  001403                            BEQ     28$             ;BR IF ALL DONE
 8080  034356  005304                            DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8081  034360  001363                            BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
 8082  034362  000760                            BR      22$             ;BR TO REPEAT THE PATTERN
 8083  034364  042737  000001  177572    28$:    BIC     #BIT0,@#SR0     ;DISABLE MEMORY MANAGEMENT
 8084  034372  000410                            BR      44$             ;GO TO RETURN
 8085                                    ;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE
 8086  034374  010003                    30$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
 8087  034376  011604                            MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8088  034400  012321                    34$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
 8089  034402  005302                            DEC     R2              ;DECREMENT WORD COUNTER
 8090  034404  001403                            BEQ     44$             ;BR IF ALL DONE
 8091  034406  005304                            DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8092  034410  001373                            BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
 8093  034412  000770                            BR      30$             ;BR TO REPEAT THE PATTERN
 8094  034414  005726                    44$:    TST     (SP)+           ;POP THE STACK
 8095  034416  104410                            RESREG                  ;RESTORE R0-R5
 8096  034420  000207                            RTS     PC              ;RETURN
 8097
 8098
```

```
                                        ;;*****************************************************************
                                        ;SBTTL  CMPBUF - SOFTWARE COMPARE DATA
                                        ;*THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
                                        ;*TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
                                        ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
                                        ;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
                                        ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
                                        ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
                                        ;;*****************************************************************
034422  104407                  CMPBUF: SAVREG                          ;SAVE R0-R5
034424  112737  000040  062045          MOVB    #40,DH701+38.           ;RESTORE ERROR MSG PARAMS
034432  012737  000007  063420          MOV     #7,DF25+2
034440  013702  005562                  MOV     WDSXFR,R2               ;SET NO. OF WORDS
034444  005037  005532                  CLR     SCRACH                  ;CLEAR COMPARE ERROR COUNT
034450  004737  041074                  JSR     PC,REPSUP               ;STORE PREV CMND FOR POSS. PRINT
034454  005737  005534                  TST     PATRN                   ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
034460  001007                          BNE     3$                      ;BR IF NOT
034462  012700  006176                  MOV     #PAT00,R0               ;GET DATA PATTERN STARTING ADDR
034466  012746  000400                  MOV     #256.,-(SP)             ;SET PATTERN WORD COUNT
034472  012701  063526                  MOV     #RWBUF,R1               ;SET BUFFER ADDRESS
034476  000466                          BR      30$                     ;PROCEED
034500  005000                  3$:     CLR     R0                      ;INIT PATTERN NO.
034502  032701  000001          4$:     BIT     #BIT0,R1                ;SEE IF THIS BIT IS SET
034506  001003                          BNE     6$                      ;BR IF THIS BIT IS SET
034510  005200                          INC     R0                      ;INCREMENT PATTERN NUMBER
034512  006201                          ASR     R1                      ;SHIFT TO EXAMINE NEXT BIT
034514  000772                          BR      4$                      ;BR TO CHECK NEXT BIT
034516  006300                  6$:     ASL     R0                      ;MULTIPLY PATTERN NO. BY 32(DEC)
034520  006300                          ASL     R0
034522  006300                          ASL     R0
034524  006300                          ASL     R0
034526  006300                          ASL     R0
034530  062700  006176                  ADD     #PAT00,R0               ;GET ADDRESS OF DESIRED PATTERN
034534  012746  000020                  MOV     #16.,-(SP)              ;PATTERN WORD COUNT
034540  013701  005572                  MOV     PMA,R1                  ;SET BUFFER ADDRESS
034544  005737  055134                  TST     $KT11                   ;SEE IF MEM MGT PRESENT
034550  100041                          BPL     30$                     ;BR IF NOT PRESENT
                                ;COMPARE LOOP FOR MEM MGT STARTS HERE
034552  013737  003176  172354          MOV     SAVPAR,@#KIPAR6         ;SET UP WORKING PAR
034560  052737  000001  177572          BIS     #BIT0,@#SR0             ;TURN ON MEM MGT
034566  042701  160000                  BIC     #160000,R1              ;FORCE RELOCATION THRU KIPAR6
034572  052701  140000                  BIS     #140000,R1
034576  010003                  22$:    MOV     R0,R3                   ;GET A COPY OF PATTERN ADDRESS
034600  011604                          MOV     (SP),R4                 ;INIT PATTERN WORD COUNT
034602  022321                  24$:    CMP     (R3)+,(R1)+             ;COMPARE DATA WORD TO PATTERN WORD
034604  001404                          BEQ     25$                     ;BR IF NO ERROR
034606  013737  172354  001216          MOV     @#KIPAR6,$REG16         ;SET PAR FOR PRINTOUT
034614  000432                          BR      35$                     ;GO HANDLE ERROR
034616  032701  020000          25$:    BIT     #BIT13,R1               ;SEE IF OVERFLOW TO NEXT PAGE
034622  001405                          BEQ     26$                     ;BR IF NO OVERFLOW
034624  062737  000200  172354          ADD     #200,@#KIPAR6           ;INCR PAR BY 4K FOR NEW PAGE
034632  042701  020000                  BIC     #BIT13,R1               ;SET PAGE = 6 AGAIN
034636  005302                  26$:    DEC     R2                      ;DECREMENT WORD COUNTER
034640  001002                          BNE     27$                     ;BR IF NOT ALL DONE YET
034642  000137  035306                  JMP     54$                     ;ALL DONE - GET OUT
```

```
 8155  034646  005304                    27$:   DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8156  034650  001354                           BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
 8157  034652  000751                           BR      22$             ;BR TO REPEAT THE PATTERN
 8158                                    ;COMPARE LOOP FOR NO MEM MGT STARTS HERE
 8159  034654  010003                    30$:   MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
 8160  034656  011604                           MOV     (SP),R4         ;INIT PATTERN WORD COUNT
 8161  034660  022321                    34$:   CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
 8162  034662  001002                           BNE     31$             ;BR IF COMPARE ERROR
 8163  034664  000137  035266                   JMP     40$             ;JUMP IF DATA COMPARES OK
 8164  034670  105037  062045              31$:   CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
 8165  034674  012737  000005  063420           MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
 8166                                    ;COMMON COMPARE ERROR HANDLER
 8167  034702  010237  001202              35$:   MOV     R2,$REG10       ;GET WORD NO.
 8168  034706  163737  005562  001202           SUB     WDSXFR,$REG10
 8169  034714  013737  001202  005570           MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
 8170  034722  004737  034016                   JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
 8171  034726  104407                           SAVREG                  ;SAVE R0-R5
 8172  034730  013700  005564                   MOV     FINCYL,R0       ;GET CYL
 8173  034734  113701  005566                   MOVB    FINTRK,R1       ;GET TRACK
 8174  034740  113702  005567                   MOVB    FINSEC,R2       ;GET SECTOR
 8175  034744  004737  037556                   JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
 8176  034750  104410                           RESREG                  ;RESTORE R0-R5
 8177  034752  032737  001000  005474           BIT     #BADSEC,RECODE
 8178  034760  001132                           BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
 8179  034762  005737  005532                   TST     SCRACH          ;CHECK THE ERROR COUNT
 8180  034766  001024                           BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
 8181  034770  105737  003134                   TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 8182  034774  001411                           BEQ     46$             ;BR IF NOT
 8183  034776  013737  005260  001174           MOV     CRMPHO,$REG5    ;GET CURRENT MAP REG 0
 8184  035004  013737  005256  001176           MOV     CRMPLO,$REG6
 8185  035012  104116                           ERROR   116             ;DATA MISCOMPARE (11/70)
 8186  035014  104117                           ERROR   117
 8187  035016  000401                           BR      48$
 8188  035020  104034                    46$:   ERROR   34              ;TYPE HEADING FOR ERROR MSG
 8189  035022  012737  177777  001174    48$:   MOV     #-1,$REG5       ;INIT CYL NO.
 8190  035030  005037  001176                   CLR     $REG6
 8191  035034  005037  001200                   CLR     $REG7
 8192  035040  005237  005532              36$:   INC     SCRACH          ;INCREMENT THE ERROR COUNT
 8193  035044  032777  000001  144066           BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
 8194  035052  001004                           BNE     39$             ;BR TO REPORT ALL ERRORS
 8195  035054  022737  000012  005532           CMP     #10.,SCRACH     ;SEE IF 10(DEC) ERRORS YET
 8196  035062  002511                           BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
 8197  035064  023737  001174  005564    38$:   CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
 8198  035072  001010                           BNE     42$             ;BR IF YES
 8199  035074  123737  001176  005566           CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
 8200  035102  001004                           BNE     42$             ;BR IF YES
 8201  035104  123737  001200  005567           CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
 8202  035112  001412                           BEQ     44$             ;BR IF SAME PACK ADDRESS
 8203  035114  013737  005564  001174    42$:   MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
 8204  035122  113737  005566  001176           MOVB    FINTRK,$REG6
 8205  035130  113737  005567  001200           MOVB    FINSEC,$REG7
 8206  035136  104115                           ERROR   115             ;TYPE NEW PACK ADDRESS
 8207  035140  005437  001202              44$:   NEG     $REG10          ;GET WORD NO.
 8208  035144  016337  177776  001204           MOV     -2(R3),$REG11   ;GET GOOD DATA
 8209  035152  016137  177776  001206           MOV     -2(R1),$REG12   ;GET BAD DATA
 8210  035160  013737  001202  001212           MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
```

```
 8211  035166  005037  001210                    CLR     $REG13
 8212  035172  006137  001212                    ROL     $REG14
 8213  035176  006137  001210                    ROL     $REG13
 8214  035202  063737  005572  001212            ADD     PMA,$REG14
 8215  035210  005537  001210                    ADC     $REG13
 8216  035214  063737  005574  001210            ADD     PMA+2,$REG13
 8217  035222  010137  001214                    MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
 8218  035226  162737  000002  001214            SUB     #2,$REG15
 8219  035234  104063                            ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
 8220  035236  032777  000100  143674            BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
 8221  035244  001020                            BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
 8222  035246  005737  005534          50$:     TST     PATRN           ;SEE IF DEFAULT DATA TEST
 8223  035252  001405                            BEQ     40$             ;BR IF YES
 8224  035254  005737  055134                    TST     $KT11           ;SEE IF MEM MGT PRESENT
 8225  035260  100002                            BPL     40$             ;BR IF NOT PRESENT
 8226  035262  000137  034616                    JMP     25$             ;GO HANDLE MEM MGT
 8227  035266  005302                  40$:      DEC     R2              ;DECREMENT WORD COUNTER
 8228  035270  001406                            BEQ     54$             ;BR IF ALL DONE
 8229  035272  005304                            DEC     R4              ;DECREMENT PATTERN WORD COUNT
 8230  035274  001002                            BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
 8231  035276  000137  034654                    JMP     30$             ;JUMP TO REPEAT THE PATTERN
 8232  035302  000137  034660          53$:     JMP     34$
 8233  035306  005737  055134          54$:     TST     $KT11           ;SEE IF MEM MGT PRESENT
 8234  035312  100003                            BPL     56$             ;BR IF NOT PRESENT
 8235  035314  042737  000001  177572            BIC     #BIT0,@#SR0     ;DISABLE MEM MGT
 8236  035322  005726                  56$:      TST     (SP)+           ;POP THE STACK
 8237  035324  104410                            RESREG                  ;RESTORE R0-R5
 8238  035326  000207                            RTS     PC              ;RETURN
 8239
 8240
 8241
 8242                                    ;;*****************************************************************
 8243                                    ;* CHKLIM - CHECK CURRENT DATA TRANSFER LIMITS
 8244                                    ;*THIS SUBROUTINE DETERMINES IF THE NEXT DATA TRANSFER SHOULD BE
 8245                                    ;*ALLOWED WITH THE CURRENT PACK ADDRESS AND WORD COUNT. IF THE
 8246                                    ;*TRANSFER WOULD CAUSE OVERFLOW BEYOND CYL 632, TRACK 1, THE
 8247                                    ;*WORD COUNT IN P.WC(R5) MUST BE SCALED DOWN TO A VALID NUMBER.
 8248                                    ;*IF NO POSSIBLE TRANSFER CAN BE ALLOWED, RETURN IS MADE TO
 8249                                    ;*ADDRESS FOLLOWING THE CALL TO CHKLIM.
 8250                                    ;*  CALL -     JSR       PC,CHKLIM
 8251                                    ;*             <"NO TRANSFER" RETURN ADDRESS>
 8252                                    ;;*****************************************************************
 8253  035330                            CHKLIM:                           ;SAVE R0-R5
 8254  035330  026527  000002  000625            CMP     P.CYLN(R5),#625 ;SEE IF SHOULD CHECK ADDR. LIMITS YET
 8255  035336  002551                            BLT     34$             ;BR TO RETURN IF NOT
 8256  035340  104407                            SAVREG                  ;SAVE R0-R5
 8257  035342  005037  003166                    CLR     SUML01
 8258  035346  005037  003170                    CLR     SUMHI1
 8259  035352  022765  000632  000002            CMP     #632,P.CYLN(R5) ;SEE IF ON LAST CYL
 8260  035360  001004                            BNE     8$              ;BR IF NOT
 8261  035362  122765  000002  000005            CMPB    #2,P.TRCK(R5)   ;SEE IF ON TRACK 2, CYL 632
 8262  035370  001537                            BEQ     36$             ;BR IF CAN'T DO TRANSFER
 8263  035372  105737  003115          8$:      TSTB    FORMAT          ;DETERMINE THE FORMAT
 8264  035376  001411                            BEQ     10$             ;BR IF 22(DEC) SECTORS
 8265                                    ;STORE CONSTANTS FOR 20(DEC) SECTORS
 8266  035400  012746  024000                    MOV     #10240.,-(SP)   ;NO. OF WORDS ON 2 TRACKS
```

```
8267  035404  012746  036000                  MOV     #15360.,-(SP)   ;NO. OF WORDS PER CY_
8268  035410  012746  012000                  MOV     #5120.,-(SP)    ;NO. OF WORDS PER TRACK
8269  035414  012746  000024                  MOV     #20.,-(SP)      ;NO. OF SECTORS
8270  035420  000410                          BR      12$
8271                                  ;STORE CONSTANTS FOR 22(DEC) SECTORS
8272  035422  012746  026000          10$:    MOV     #11264.,-(SP)   ;NO. OF WORDS ON 2 TRACKS
8273  035426  012746  041000                  MOV     #16896.,-(SP)   ;NO. OF WORDS PER CYL
8274  035432  012746  013000                  MOV     #5632.,-(SP)    ;NO. OF WORDS PER TRACK
8275  035436  012746  000026                  MOV     #22.,-(SP)      ;NO. OF SECTORS
8276                                  ;COMPUTE NO. OF WORDS LEFT ON THIS TRACK
8277  035442  012603                  12$:    MOV     (SP)+,R3        ;GET NO. OF SECTORS
8278  035444  116502  000004                  MOVB    P.SECT(R5),R2   ;GET CURRENT SECTOR NO.
8279  035450  160203                          SUB     R2,R3           ;NO. OF SECTORS LEFT
8280  035452  005002                          CLR     R2              ;GET NO. OF WDS IN THESE SECTORS
8281  035454  012705  000400                  MOV     #256.,R5
8282  035460  005004                          CLR     R4
8283  035462  004737  052422                  JSR     PC,M.DPIM
8284  035466  010337  003166                  MOV     R3,SUML01       ;STORE THIS NO.
8285  035472  016605  000012                  MOV     12(SP),R5       ;RESTORE PARAM BLK ADDR
8286                                  ;COMPUTE NO. OF WORDS LEFT ON THIS CYLINDER
8287  035476  012703  000002                  MOV     #2,R3           ;TRACK LIMIT = 2
8288  035502  022765  000632  000002          CMP     #632,P.CYLN(R5) ;SEE IF ON CYL 632
8289  035510  001001                          BNE     14$             ;BR IF NOT
8290  035512  005303                          DEC     R3              ;DECREMENT TRACK LIMIT TO 1 FOR CYL 632
8291  035514  116502  000005          14$:    MOVB    P.TRCK(R5),R2   ;GET CURRENT TRACK NO.
8292  035520  160203                          SUB     R2,R3           ;GET NO. OF TRACKS LEFT
8293  035522  005002                          CLR     R2              ;GET NO. OF WORDS IN THESE TRACKS
8294  035524  012605                          MOV     (SP)+,R5        ;NO. OF WDS PER TRACK
8295  035526  004737  052422                  JSR     PC,M.DPIM
8296  035532  060337  003166                  ADD     R3,SUML01       ;ADD WORDS TO TOTAL
8297  035536  016605  000010                  MOV     10(SP),R5       ;RESTORE PARAM BLK ADR
8298                                  ;COMPUTE NO. OF WORDS ON WHOLE CYLINDERS REMAINING
8299  035542  012703  000631                  MOV     #631,R3         ;CYL LIMIT =631
8300  035546  166503  000002                  SUB     P.CYLN(R5),R3   ;GET NO. OF WHOLE CYLS LEFT
8301  035552  100001                          BPL     16$
8302  035554  005003                          CLR     R3
8303  035556  005002                  16$:    CLR     R2
8304  035560  012605                          MOV     (SP)+,R5        ;GET NO. OF WDS PER CYL
8305  035562  004737  052422                  JSR     PC,M.DPIM       ;COMPUTE NO. OF WDS IN THESE WHOLE CYLS
8306  035566  063703  003166                  ADD     SUML01,R3       ;ADD THESE WDS TO TOTAL
8307  035572  005502                          ADC     R2
8308  035574  063702  003170                  ADD     SUMHI1,R2
8309  035600  016605  000006                  MOV     6(SP),R5        ;RESTORE PARAM BLK ADDR
8310                                  ;ADD THE NO. OF WDS ON TRACKS 0,1, CYL 632
8311  035604  022765  000632  000002          CMP     #632,P.CYLN(R5) ;SEE IF CYL = 632
8312  035612  001002                          BNE     18$             ;BR IF NOT 632
8313  035614  005726                          TST     (SP)+           ;POP STACK
8314  035616  000402                          BR      20$
8315  035620  062603                  18$:    ADD     (SP)+,R3        ;ADD WDS ON TRKS 0,1, CYL 632
8316  035622  005502                          ADC     R2
8317                                  ;GET DESIRED WORD COUNT
8318  035624  005000                  20$:    CLR     R0              ;WORD COUNT HI BITS
8319  035626  016501  000012                  MOV     P.WC(R5),R1     ;GET THE DESIRED WORD COUNT
8320  035632  005401                          NEG     R1
8321  035634  001001                          BNE     22$             ;BR IF WORD COUNT NOT 65,536(DEC)
8322  035636  005200                          INC     R0              ;SET HI WORD COUNT = 1
```

```
                                ;SUBTRACT WORD COUNT FROM NO. OF WORDS LEFT TO DETERMINE IF OVERFLOW.
                                ;   NUMBER OF WORDS LEFT IS IN R2-R3, WORD COUNT IS IN R0-R1.
035640  160103                  22$:    SUB     R1,R3           ;LO PARTS
035642  005602                          SBC     R2
035644  160002                          SUB     R0,R2           ;HI PARTS
035646  100004                          BPL     30$             ;BR IF WORD COUNT NOT > NO. OF WDS LEFT
                                ;SCALE DOWN WORD COUNT TO AVOID OVERFLOW (SUBTRACT THE EXCESS)
035650  060301                          ADD     R3,R1           ;GET NEW WORD COUNT IN R1
035652  005401                          NEG     R1
035654  010165  000012                  MOV     R1,P.WC(R5)     ;SET NEW WORD COUNT IN PARAM BLOCK
                                ;RETURN
035660  104410                  30$:    RESREG                  ;RESTORE R0-R5
035662  062716  000002          34$:    ADD     #2,(SP)         ;FIX NORMAL RETURN PC
035666  000207                          RTS     PC              ;EXIT
035670  017616  000000          36$:    MOV     @(SP),(SP)      ;FIX "NO TRANSFER" RETURN PC
035674  104410                          RESREG                  ;RESTORE R0-R5
035676  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (↑C) OR (↑Z) TTY INPUT.
                                ;*IF (↑C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (↑Z)
                                ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
                                ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
                                ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (↑C) OR (↑Z),
                                ;*NO ACTION IS TAKEN.
                                ;*  CALL -   JSR     PC,CTLOUT
                                ;*    OR -   CKEXIT
                                ;;*****************************************************************
035700  122737  000001  003110  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
035706  001077                          BNE     10$             ;BR IF NOT RUNNING TESTS
035710  005737  005522                  TST     INTCHR          ;SEE IF ANY TTY INPUT
035714  001474                          BEQ     10$             ;BR IF NO INPUT
035716  105737  003106                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
035722  001446                          BEQ     12$             ;BR IF YES
035724  122737  000003  005522          CMPB    #003,INTCHR     ;SEE IF (↑C) TYPED
035732  001033                          BNE     4$              ;BR IF NOT (↑C)
035734  012700  013660                  MOV     #DRVTST,R0      ;SET RETURN ADDR = DRVTST
035740  105737  003124          2$:     TSTB    XOVLAD          ;SEE IF XXDP CURRENTLY OVERLAID
035744  001402                          BEQ     6$              ;BR IF NOT
035746  004737  027040                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP, IF NECESSARY
035752  000005                  6$:     RESET                   ;RESET ALL DEVICES
035754  005037  005522                  CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
035760  005037  001304                  CLR     $TIMES          ;CLEAR THE ITER. COUNT
035764  105037  003125                  CLRB    XDPSVD          ;CLEAR THE XXDP SAVED FLAG
035770  012737  041354  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADDRESS
035776  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK
036002  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
036010  004737  037662                  JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
036014  005037  001102                  CLR     $TSTNM          ;CLEAR THE TEST NO.
036020  000110                          JMP     @R0             ;EXIT FROM TESTS
036022  122737  000032  005522  4$:     CMPB    #032,INTCHR     ;SEE IF (↑Z) TYPED
036030  001016                          BNE     7$              ;BR IF NOT (↑Z)
036032  012700  015300                  MOV     #INPUTP,R0      ;SET RETUR', ADDR = INPUTP
036036  000740                          BR      2$              ;TAKE EXIT
```

```
8379  036040  122737  000003  005522  12$:    CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
8380  036046  001007                          BNE     7$              ;BR IF NOT
8381  036050  104401  010024                  TYPE    ,HLTRQD         ;TYPE "HALT REQUESTED"
8382  036054  104401  007774                  TYPE    ,CNTRDY         ;TYPE "PRESS CONT WHEN RDY"
8383  036060  012700  043364                  MOV     #HLTPRG,R0      ;SET HALT ADDRESS
8384  036064  000726                          BR      2$              ;TAKE EXIT
8385  036066  122737  000007  005522  7$:     CMPB    #007,INTCHR     ;SEE IF (^G) TYPED
8386  036074  001003                          BNE     8$              ;BR IF NOT (^G)
8387  036076  004737  025642                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
8388  036102  004737  025330          8$:     JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
8389  036106  000207                  10$:    RTS     PC              ;RETURN
8390
8391
8392
8393
8394
8395
8396
8397
8398                                          ;;*****************************************************************
8399                                          ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS OUT TO DISK AND PERFORM
8400                                          ;*A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
8401                                          ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
8402                                          ;*USED.
8403                                          ;;*****************************************************************
8404  036110  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
8405                                  ;LOAD THE R/W BUFFER WITH DATA
8406  036112  012700  063526                  MOV     #RWBUF,R0       ;BUFFER ADDRESS
8407  036116  012701  000400                  MOV     #400,R1         ;400(OCT) WORDS
8408  036122  010320                  4$:     MOV     R3,(R0)+        ;LOAD A BUFFER WORD
8409  036124  005301                          DEC     R1              ;DECR COUNTER
8410  036126  001375                          BNE     4$              ;BR IF NOT DONE YET
8411                                  ;SET UP PARAMETERS
8412  036130  012765  063526  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
8413  036136  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
8414                                  ;WRITE THE DATA
8415  036144  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
8416  036152  004737  037662                  JSR     PC,DRVCAL       ;WRITE THE DATA
8417                                  ;PERFORM WRITE CHECK
8418  036156  112765  000131  000001  6$:     MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
8419  036164  004737  037662                  JSR     PC,DRVCAL       ;PERFORM WRITE CHECK
8420  036170  104410                          RESREG                  ;RESTORE R0-R5
8421  036172  000207                          RTS     PC              ;RETURN
8422
8423
8424
8425                                          ;;*****************************************************************
8426                                          ;*INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
8427                                          ;;*****************************************************************
8428  036174  104407                  INCRMA: SAVREG                  ;SAVE R0-R5
8429  036176  005001                          CLR     R1
8430  036200  016500  000012                  MOV     P.WC(R5),R0     ;GET WORD COUNT
8431  036204  001002                          BNE     4$              ;BR IF NOT 65,536(DEC)
8432  036206  005201                          INC     R1              ;SET HI BIT
8433  036210  000401                          BR      6$              ;CONTINUE
8434  036212  005400                  4$:     NEG     R0              ;GET TRUE WORD COUNT
8435  036214  000241                  6$:     CLC                     ;DOUBLE THE WORD COUNT TO GET BYTES
8436  036216  006100                          ROL     R0
8437  036220  006101                          ROL     R1
8438  036222  060037  005742                  ADD     R0,MA           ;ADD IT TO COMPUTE NEW MA
8439  036226  005537  005744                  ADC     MA+2
```

```
                        ADD     R1,MA+2
                        RESPEC                  ;RESTORE R0-R5
                        RTS     PC              ;RETURN


                ;;*******************************************************************
                ;*RECBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
                ;*INTO BSSOFT.
                ;;*******************************************************************
                RECBSF:
                        JSR     PC,INITSS       ;INIT THE S.S.
                        CLRB    P.CS1H(R5)      ;SET 22-SECTOR FORMAT
                        MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
                        MOV     #632,P.CYLN(R5) ;SET CYL = 632
                        MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
                        MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
                        TSTB    FORMAT
                        BEQ     6$              ;BR IF 22 SECTORS
                        INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
                        INC     R3
                6$:     MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
                        MOV     #BSFACT,P.BAL0(R5) ;SET BA FOR FACTORY BSF
                8$:     MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
                        CLRB    WCEFLG          ;INIT THE ERROR FLAG
                        JSR     PC,DRVCAL       ;READ THE FACTORY BSF
                        TSTB    WCEFLG          ;SEE IF ANY ERRORS
                        BEQ     12$             ;BR IF NOT
                        ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
                        CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
                        BNE     8$              ;BR IF NOT YET
                10$:    JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
                        ERROR   120             ;ABORTING- BAD BSF READ
                        JMP     HLTPRG          ;HALT- CAN'T PROCEED
                12$:    MOV     #BSSOFT,P.BAL0(R5) ;SET BA FOR SOFTWARE RECORD
                        MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
                        MOV     #22.,R3         ;SET 22 SECTOR LIMIT
                        TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
                        BEQ     14$             ;BR IF YES
                        MOV     #19.,R3         ;SET 20 SECTOR LIMIT
                14$:    MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
                        CLRB    WCEFLG          ;INIT THE ERROR FLAG
                        JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
                        TSTB    WCEFLG          ;SEE IF ANY ERRORS
                        BEQ     16$             ;BR IF NOT
                        ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
                        CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
                        BLE     14$             ;BR IF NOT YET
                        BR      10$             ;REPORT ERROR AND ABORT
                16$:    MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
                        RTS     PC              ;RETURN

                ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
                ;*POSSIBLE ERRORS IN READING BAD SECTORS.
                BDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
                        BNE     4$              ;BR IF A READ ERROR OCCURRED
```

```
                                        JMP     ERRHDL          ;GO HANDLE OTHER ERROR
                                4$:     INCB    WCEFLG          ;SET ERROR FLAG
                                        JMP     RETNML          ;TAKE NORMAL DRIVER RETURN


                                ;;*****************************************************************
                                ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
                                ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
                                ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
                                ;*2 BSF'S.
                                ;;*****************************************************************
                                TYPBSF: SAVREG                  ;SAVE R0-R5
                                        CLR     R1              ;INIT LIMIT COUNTER
                                        CLR     R4              ;INIT BSF FILE INDICATOR
                                        MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
                                        TYPE    .FACTBS         ;TYPE "FACTORY"
                                        TYPE    .BDSECT         ;TYPE "BAD SECTORS"
                                3$:     TYPE    .DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
                                        TYPE    .$CRLF
                                4$:     TST     (R0)            ;SEE IF A SECTOR LISTED HERE
                                        BPL     8$              ;BR IF YES
                                        TST     R1              ;SEE IF FILE IS EMPTY
                                        BNE     14$             ;BR IF NOT EMPTY
                                        TYPE    .NOFALS         ;TYPE "NONE"
                                        TYPE    .$CRLF          ;TYPE <CR>,<LF>
                                        BR      14$
                                8$:     MOV     (R0)+,-(SP)     ;GET A CYL NO.
                                        TYPOC                   ;TYPE IT
                                        TYPE    .SPACE2         ;TYPE SEPARATORS
                                        MOV     (R0),R3         ;GET TRACK AND SECTOR
                                        CLRB    R3
                                        SWAB    R3              ;GET THE TRACK NO.
                                        MOV     R3,-(SP)
                                        TYPOC                   ;TYPE IT
                                        TYPE    .SPACE2         ;TYPE SEPARATORS
                                        MOVB    (R0),R3         ;GET SECTOR NO.
                                        MOV     R3,-(SP)
                                        TYPOC                   ;TYPE IT
                                        TYPE    .$CRLF
                                        TST     (R0)+           ;INCR THE POINTER
                                        INC     R1              ;INCR THE COUNTER
                                        CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
                                        BLT     4$              ;BR IF NOT YET
                                14$:    TST     R4              ;SEE IF BOTH FILES TYPED YET
                                        BNE     18$             ;BR IF NOT YET
                                        INC     R4              ;SET INDICATOR FOR SOFT. FILE
                                        MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
                                        TYPE    .SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
                                        BR      3$              ;GO TYPE SOFT. BSF
                                18$:    RESREG                  ;RESTORE R0-R5
                                        RTS     PC              ;RETURN


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

```
                                        ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
                                        ;*TERMINATED BY BSE ERROR , AND STORE IT IN WDSXFR.
                                        ;;*****************************************************************
036702  104407                  FINMEM: SAVREG                  ;SAVE R0-R5
036704  016500  000002                  MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
036710  116501  000005                  MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
036714  116502  000004                  MOVB    P.SECT(R5),R2   ;SECTOR NO.
036720  005003                          CLR     R3              ;INIT SECTOR COUNT TO 0
036722  026500  000030          6$:     CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
036726  001006                          BNE     8$              ;BR IF NOT EQUAL
036730  126501  000027                  CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
036734  001003                          BNE     8$              ;BR IF NOT EQUAL
036736  126502  000026                  CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
036742  001404                          BEQ     12$             ;BR IF ADRS ARE EQUAL
036744  005203                  8$:     INC     R3              ;INCR SECTOR COUNT
036746  004737  036766                  JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
036752  000763                          BR      6$              ;GO COMPARE ADDRESSES
036754  000303                  12$:    SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
036756  010337  005562                  MOV     R3,WDSXFR       ;STORE IT
036762  104410                          RESREG                  ;RESTORE R0-R5
036764  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
                                        ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
                                        ;*IN R2.
                                        ;;*****************************************************************
036766  012704  000025          INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
036772  105737  003115                  TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
036776  001402                          BEQ     4$              ;BR IF YES
037000  012704  000023                  MOV     #19.,R4         ;SET SECTOR LIMIT
037004  005202                  4$:     INC     R2              ;INCR. SECTOR NO.
037006  020204                          CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
037010  003407                          BLE     6$              ;BR IF NOT
037012  005002                          CLR     R2              ;SET SECTOR = 0
037014  005201                          INC     R1              ;INCR. TRACK NO.
037016  020127  000002                  CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
037022  003402                          BLE     6$              ;BR IF NOT
037024  005001                          CLR     R1              ;SET TRACK = 0
037026  005200                          INC     R0              ;INCR. CYLINDER NO.
037030  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).
                                        ;;*****************************************************************
037032  104407                  MIDXFR: SAVREG                  ;SAVE R0-R5
037034  004737  036702                  JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
037040  016500  000030                  MOV     P.DCYL(R5),R0   ;GET CYL NO.
037044  116501  000027                  MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
```

```
037050  116502  000026                  MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
037054  004737  036766                  JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
037060  010065  000002                  MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
037064  110165  000005                  MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
037070  110265  000004                  MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
037074  013703  005562                  MOV     WDSXFR,R3       ;GET NO. OF WORDS XFERRED
037100  062703  000400                  ADD     #400,R3         ;SKIP BAD SECTOR
037104  060365  000012                  ADD     R3,P.WC(R5)     ;UPDATE P.WC(R5)
037110  005004                          CLR     R4              ;GET BYTES XFERRED
037112  006103                          ROL     R3
037114  006104                          ROL     R4
037116  060337  005572                  ADD     R3,PMA          ;UPDATE PMA,PMA+2
037122  005537  005574                  ADC     PMA+2
037126  060437  005574                  ADD     R4,PMA+2
037132  013765  005572  000010          MOV     PMA,P.BALO(R5)  ;UPDATE P.BALO(R5)
037140  013700  005574                  MOV     PMA+2,R0
037144  042700  177774                  BIC     #177774,R0
037150  150065  000007                  BISB    R0,P.BAHI(R5)   ;UPDATE P.BAHI(R5)
037154  005737  055134                  TST     $KT11           ;SEE IF MEM MGT
037160  100002                          BPL     16$
037162  004737  026362                  JSR     PC,PREPAR       ;UPDATE MEM MGT AND UNIBUS MAP
037166  104410          16$:    RESREG                          ;RESTORE R0-R5
037170  000207                  RTS     PC                      ;RETURN


;;**********************************************************************
;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
;;**********************************************************  .******
037172  104407          SVPRMS: SAVREG                          ;SAVE R0-R5
037174  012700  002622          MOV     #PARM0+2,R0             ;ADRS OF PARAMS
037200  012701  005550          MOV     #SAVPRS,R1              ;ADRS OF SAVE AREA
037204  016537  000012  005562  MOV     P.WC(R5),WDSXFR         ;GET WORD COUNT
037212  005437  005562          NEG     WDSXFR                  ;MAKE IT POSITIVE
037216  012737  063526  005572  MOV     #RWBUF,PMA              ;INIT PMA TO RWBUF
037224  005037  005574          CLR     PMA+2                   ;INIT PMA+2 TO 0
037230  005737  005534          TST     PATRN                   ;SEE IF DEFAULT DATA TEST
037234  001414                  BEQ     GTO                     ;BR IF YES
037236  013737  005742  005572  MOV     MA,PMA                  ;SET PMA=MA
037244  013737  005744  005574  MOV     MA+2,PMA+2              ;SET PMA+2=MA+2
037252  000405                  BR      GTO
037254  104407          GTPRMS: SAVREG                          ;SAVE R0-R5
037256  012700  005550          MOV     #SAVPRS,R0              ;ADRS OF SAVE AREA
037262  012701  002622          MOV     #PARM0+2,R1             ;ADRS OF PARAMS
037266  012702  000005  GTO:    MOV     #5,R2                   ;SET FOR 5 WORDS
037272  012021          6$:     MOV     (R0)+,(R1)+             ;MOVE A WORD
037274  005302                  DEC     R2                      ;SEE IF DONE YET
037276  001375                  BNE     6$                      ;BR IF NOT YET
037300  104410                  RESREG                          ;RESTORE R0-R5
037302  000207                  RTS     PC                      ;RETURN


;;**********************************************************************
;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS. IF
```

```
                                        ;*ENCOUNTERED. THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
                                        ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED.
                                        ;;*****************************************************************
037304  010446                  $RNSFR: MOV     R4,-(SP)        ;SAVE R4 ON STACK
037306  005004                          CLR     R4              ;CLEAR BSE ERROR INDICATOR
037310  105037  003131                  CLRB    WCEFLG          ;INIT WCE ERROR FLAG TO 0
037314  004737  037662            4$:   JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
037320  032737  000100  005474        BIT     #WCERR,RECODE   ;SEE IF WCE ERROR OCCURRED
037326  001403                          BEQ     5$              ;BR IF NOT
037330  112737  000001  003131        MOVB    #1,WCEFLG       ;SET WCE ERROR FLAG
037336  032737  000002  005474  5$:   BIT     #BSERR,RECODE   ;SEE IF BSE ERROR OCCURRED
037344  001406                          BEQ     6$              ;BR IF NOT
037346  005204                          INC     R4              ;INCR BSE ERROR INDICATOR
037350  004737  037032                  JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
037354  005765  000012                  TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
037360  001355                          BNE     4$              ;BR IF NOT
037362  005704                  6$:     TST     R4              ;SEE IF ANY BSE ERRORS OCCURRED
037364  001411                          BEQ     8$              ;BR IF NOT
037366  004737  037254                  JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
037372  004737  037172                  JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
037376  005737  055134                  TST     $KT11           ;SEE IF MEM MGT PRESENT
037402  100002                          BPL     8$              ;BR IF NOT
037404  004737  026362                  JSR     PC,PREPAR       ;PREPARE MEM MGT AND U.M.
037410  012604                  8$:     MOV     (SP)+,R4        ;RESTORE R4
037412  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        ;SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
                                        ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
                                        ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
                                        ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
                                        ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
                                        ;*
                                        ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
                                        ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
                                        ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
                                        ;;*****************************************************************
037414  010237  001266          $RHTBS: MOV     R2,$TMP2
037420  010337  001270                  MOV     R3,$TMP3
037424  012637  001272                  MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
037430  011402                          MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
037432  012437  001264                  MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
037436  062737  001000  001264          ADD     #1000,$TMP1
037444  005003                          CLR     R3              ;CLEAR R3 FOR COUNT
037446  062702  000010                  ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
037452  005712                  1$:     TST     (R2)            ;TEST IF ALL ONES
037454  100430                          BMI     5$              ;YES-DONE
037456  020012                          CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
037460  001406                          BEQ     3$              ;YES-GO CHECK TRACK
037462  062702  000004                  ADD     #4,R2           ;ELSE BUMP POINTER
037466  020237  001264                  CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
037472  002021                          BGE     5$              ;YES-EXIT
037474  000766                          BR      1$              ;LOOP
037476  005722                  3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
```

```
 8715  037500  011237  001302                MOV     (R2),$TMP10     ;GET TRK/SEC WORD
 8716  037504  042737  174377  001302        BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
 8717  037512  123701  001303                CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
 8718  037516  001402                        BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
 8719  037520  005722                        TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
 8720  037522  000753                        BR      1$              ;GO TEST NEXT CYL
 8721  037524  005203                4$:     INC     R3              ;BUMP BAD SECTOR COUNT
 8722  037526  012246                        MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
 8723  037530  042716  177700                BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
 8724  037534  000746                        BR      1$              ;GO CHECK REST OF FILE
 8725  037536  010346                5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
 8726  037540  013702  001266                MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
 8727  037544  013703  001270                MOV     $TMP3,R3        ;REGISTERS
 8728  037550  013746  001272                MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
 8729  037554  000204                        RTS     R4              ;RETURN
 8730                                ;;*****************************************************************
 8731                                .SBTTL  BAD SECTOR CHECK ROUTINE
 8732                                ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
 8733                                ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
 8734                                ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
 8735                                ;*IS NOT LISTED THE FLAG IS RESET.
 8736                                ;;*****************************************************************
 8737  037556  104407                BDSRCK: SAVREG
 8738  037560  012737  004222  037572        MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
 8739  037566  004437  037414        2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
 8740  037572  000000                1$:     .WORD                   ;TABLE ADDRESS GOES HERE
 8741  037574  012603                        MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
 8742  037576  001015                        BNE     6$              ;IF ANY, GO TEST WHICH ONES
 8743  037600  023727  037572  003222  7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
 8744  037606  001404                        BEQ     3$              ;IF YES-EXIT
 8745  037610  012737  003222  037572        MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
 8746  037616  000763                        BR      2$              ;GO SEARCH IT
 8747  037620  042737  001000  005474  3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
 8748  037626  104410                4$:     RESREG
 8749  037630  000207                        RTS     PC              ;RETURN
 8750  037632  022602                6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
 8751  037634  001403                        BEQ     8$              ;YES-GO SET BIT & EXIT
 8752  037636  005303                        DEC     R3              ;DECREMENT BAD SECTOR COUNT
 8753  037640  001374                        BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
 8754  037642  000756                        BR      7$              ;ELSE GO SEARCH OTHER TABLE
 8755  037644  052737  001000  005474  8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
 8756  037652  005303                9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
 8757  037654  001764                        BEQ     4$              ;NUMBER
 8758  037656  005726                        TST     (SP)+
 8759  037660  000774                        BR      9$              ;EXIT
 8760
 8761
 8762
 8763                                ;;*****************************************************************
 8764                                .SBTTL  CALL DRIVER ROUTINE
 8765                                ;*ENTRY JSR PC,DRVCAL
 8766                                ;*                  WITH R5 POINTING TO PARAMETER BLOCK
 8767                                ;*RETURN RTS PC
 8768                                ;*
 8769                                ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
 8770                                ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
```

```
 8771                                   ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
 8772                                   ;*BLOCK WHEN THE ROUTINE IS CALLED.
 8773                                   ;*
 8774                                   ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
 8775                                   ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
 8776                                   ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
 8777                                   ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
 8778                                   ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
 8779                                   ;;*****************************************************************
 8780  037662                           DRVCAL:
 8781  037662  004737  032040                  JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
 8782  037666  004737  035700                  JSR     PC,CTLOUT       ;CHECK FOR (↑C) OR (↑Z) KBD INPUT
 8783  037672  105037  003113                  CLRB    DONE            ;CLEAR DONE FLAG
 8784  037676  105037  003127                  CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
 8785  037702  004737  037776                  JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
 8786  037706  010537  037716                  MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
 8787  037712  004737  050006                  JSR     PC,C.INIT       ;CALL DRIVER
 8788  037716  000000           4$:     .WORD                           ;P.B. ADDRESS GOES HERE
 8789  037720  004737  044356           6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
 8790  037724  105737  003113                  TSTB    DONE            ;DONE SET?
 8791  037730  001773                          BEQ     6$              ;NO-LOOP
 8792  037732  105737  003127                  TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
 8793  037736  001416                          BEQ     12$             ;BR IF NOT
 8794  037740  105037  003113                  CLRB    DONE            ;CLEAR DONE FLAG
 8795  037744  105037  003127                  CLRB    DRNAFG          ;CLEAR DRIVE NOT AVAIL FLAG
 8796  037750  010537  037760                  MOV     R5,8$           ;GET PARAMETER BLOCK ADDRESS
 8797  037754  004737  050006                  JSR     PC,C.INIT       ;RE-ISSUE THE COMMAND
 8798  037760  000000           8$:     .WORD                           ;P.B. ADDRESS GOES HERE
 8799  037762  004737  044356           10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG
 8800  037766  105737  003113                  TSTB    DONE            ;DONE SET ?
 8801  037772  001773                          BEQ     10$             ;NO - LOOP
 8802  037774  000207           12$:    RTS     PC                      ;YES-RETURN
 8803
 8804
 8805
 8806                                   ;;*****************************************************************
 8807                                   ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
 8808                                   ;;*****************************************************************
 8809  037776  104407           STRCMD: SAVREG                          ;SAVE R0-R5
 8810  040000  012701  005222                  MOV     #PRVCMD,R1      ;ADDR OF PREV COMMAND STORAGE
 8811  040004  012700  005236                  MOV     #COMSTR,R0      ;ADDR OF CURRENT COMMAND STORAGE
 8812                                   ;STORE PREVIOUS COMMAND
 8813  040010  012021                          MOV     (R0)+,(R1)+
 8814  040012  012021                          MOV     (R0)+,(R1)+
 8815  040014  012021                          MOV     (R0)+,(R1)+
 8816  040016  012021                          MOV     (R0)+,(R1)+
 8817  040020  012021                          MOV     (R0)+,(R1)+
 8818  040022  012021                          MOV     (R0)+,(R1)+
 8819  040024  105737  003134                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 8820  040030  001414                          BEQ     4$              ;BR IF NOT
 8821  040032  013737  005260  005254          MOV     CRMPH0,PRMPH0   ;STORE PREV U.B. MAP REG 0
 8822  040040  013737  005256  005252          MOV     CRMPL0,PRMPL0
 8823  040046  013737  170202  005260          MOV     @#MAPH00,CRMPH0 ;STORE CURRENT U.B. MAP REG 0
 8824  040054  013737  170200  005256          MOV     @#MAPL00,CRMPL0
 8825                                   ;STORE CURRENT COMMAND
 8826  040062  012701  005236           4$:     MOV     #COMSTR,R1
```

```
 8827  040066  012521                    MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BL
 8828  040070  012521                    MOV     (R5)+,(R1)+
 8829  040072  012521                    MOV     (R5)+,(R1)+
 8830  040074  012521                    MOV     (R5)+,(R1)+
 8831  040076  012521                    MOV     (R5)+,(R1)+
 8832  040100  012521                    MOV     (R5)+,(R1)+
 8833  040102  104410                    RESREG                  ;RESTORE R0-R5
 8834  040104  000207                    RTS     PC              ;RETURN
 8835
 8836
 8837
 8838                            ;;*****************************************************************
 8839                            .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
 8840                            ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
 8841                            ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
 8842                            ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
 8843                            ;*ROUTINE.
 8844                            ;;*****************************************************************
 8845  040106  152737  000377  003113  ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
 8846  040114  032737  100000  005474          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
 8847  040122  001403                          BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
 8848  040124  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 8849  040130  000413                          BR      1$              ;EXIT
 8850  040132  105737  003116          2$:     TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
 8851  040136  001410                          BEQ     1$              ;NO - SKIP TO EXIT
 8852  040140  005037  001174                  CLR     $REGS
 8853  040144  113737  003116  001174          MOVB    ERRCNT,$REGS    ;GET RETRY COUNT
 8854  040152  104101                          ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
 8855  040154  105037  003116                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 8856  040160  005037  005474          1$:     CLR     RECODE          ;CLEAR RECOVERY FLAGS
 8857  040164  000207                          RTS     PC              ;RETURN
 8858                            ;;*****************************************************************
 8859                            .SBTTL  TYPE ERROR ROUTINE
 8860                            ;*ENTRY -       JSR     PC,TYPERR
 8861                            ;*RETURN -      RTS     PC
 8862                            ;*
 8863                            ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 8864                            ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
 8865                            ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
 8866                            ;*THE ERROR.
 8867                            ;;*****************************************************************
 8868  040166  104407            TYPERR: SAVREG
 8869  040170  105237  003120                  INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
 8870  040174  032737  000040  005750          BIT     #BIT5,CS        ;SEE IF DRIVE SHOULD BE DROPPED
 8871                                                                  ;  IF ERROR LIMIT EXCEEDED
 8872  040202  001412                          BEQ     9$              ;BR IF NOT
 8873  040204  123727  003120  000024          CMPB    DRVERS,#20.     ;SEE IF 20(DEC) ERRORS EXCEEDED
 8874  040212  002406                          BLT     9$              ;BR IF NOT
 8875  040214  105037  003120                  CLRB    DRVERS          ;CLEAR DRIVE ERROR COUNT
 8876  040220  104401  011255                  TYPE    ,DROPDR         ;TYPE "DROPPING DRIVE"
 8877  040224  000137  016772                  JMP     NEWDRV          ;PROCEED TO TEST NEXT DRIVE
 8878  040230  032777  020000  140702  9$:     BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
 8879  040236  001402                          BEQ     6$              ;BR IF NO
 8880  040240  000137  040644                  JMP     20$
 8881  040244  005000                  6$:     CLR     R0              ;CLR R0 FOR ERROR NUMBER
 8882  040246  005005                          CLR     R5              ;INIT INDENT INDICATOR
```

```
 8883  040250  005105                          COM     R5
 8884  040252  113700  001114                  MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
 8885  040256  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
 8886  040260  006300                          ASL     R0
 8887  040262  006300                          ASL     R0
 8888  040264  006300                          ASL     R0
 8889  040266  062700  001400          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
 8890  040272  012037  040312                  MOV     (R0)+,2$        ;GET EM POINTER
 8891  040276  001406                          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 8892  040300  104401  012413                  TYPE    .CR2LF
 8893  040304  104401  055572                  TYPE    ,AS2SP2         ;TYPE "**  "
 8894  040310  104401                          TYPE                    ;TYPE ERROR MESSAGE (EM)
 8895  040312  000000          2$:     .WORD   0                       ;EM POINTER GOES HERE
 8896  040314  012037  040470          3$:     MOV     (R0)+,4$        ;GET DH POINTER
 8897  040320  001467                          BEQ     5$              ;BR IF THERE ISN'T ONE
 8898  040322  104401  001315                  TYPE    ,$CRLF
 8899  040326  104401  055560                  TYPE    ,TSTMSG         ;TYPE "    TEST "
 8900  040332  013746  001102                  MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
 8901  040336  104403                          TYPOS                   ;TYPE IT
 8902  040340     002                          .BYTE   2               ;2 DIGITS
 8903  040341     000                          .BYTE   0               ;SUPPRESS LEADING ZEROS
 8904  040342  104401  012413                  TYPE    .CR2LF
 8905  040346  032777  010000  140564          BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 8906  040354  001133                          BNE     20$             ;BR IF YES
 8907  040356  104401  060714                  TYPE    .DH105          ;TYPE "PREVIOUS COMMAND :"
 8908  040362  104401  001315                  TYPE    ,$CRLF
 8909  040366  104401  061106                  TYPE    ,DH101+10       ;TYPE PREV COMMAND HEADER
 8910  040372  104401  001315                  TYPE    ,$CRLF
 8911  040376  012701  000006                  MOV     #6,R1           ;SIX COMMAND VALUES
 8912  040402  012702  001236                  MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
 8913  040406  012246          30$:    MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
 8914  040410  104402                          TYPOC                   ;TYPE IT
 8915  040412  104401  012427                  TYPE    ,SPACE2         ;TYPE SEPARATORS
 8916  040416  005301                          DEC     R1              ;SEE IF 7 VALUES TYPED YET
 8917  040420  001372                          BNE     30$             ;BR IF NOT
 8918  040422  104401  001315                  TYPE    ,$CRLF
 8919  040426  104401  012427                  TYPE    ,SPACE2         ;INDENT
 8920  040432  104401  061165                  TYPE    ,DH102          ;TYPE HDR FOR BA DATA
 8921  040436  104401  001315                  TYPE    ,$CRLF
 8922  040442  104401  012427                  TYPE    ,SPACE2         ;INDENT
 8923  040446  012246                          MOV     (R2)+,-(SP)
 8924  040450  104402                          TYPOC                   ;TYPE PREV. HI BA BITS
 8925  040452  104401  012427                  TYPE    ,SPACE2
 8926  040456  011246                          MOV     (R2),-(SP)
 8927  040460  104402                          TYPOC                   ;TYPE PREV. LO BA BITS
 8928  040462  104401  012413                  TYPE    ,CR2LF
 8929  040466  104401                          TYPE                    ;TYPE DATA HEADER
 8930  040470  000000          4$:     .WORD   0                       ;DH POINTER GOES HERE
 8931  040472  104401  001315                  TYPE    ,$CRLF
 8932  040476  005005                          CLR     R5              ;INIT INDENT INDICATOR
 8933  040500  032777  010000  140432  5$:     BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 8934  040506  001056                          BNE     20$             ;BR IF YES
 8935  040510  012001                          MOV     (R0)+,R1        ;GET DT POINTER
 8936  040512  001454                          BEQ     20$             ;BRANCH IF THERE ARE NONE
 8937  040514  012000                          MOV     (R0)+,R0        ;GET DF POINTER
 8938  040516  012002                          MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
```

```
8939  040520  112003                10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
8940  040522  105720                        TSTB    (R0)+           ;BUMP PAST FORMAT WORD
8941  040524  005703                        TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
8942  040526  001417                        BEQ     14$             ;NO - SKIP DATA PRINT
8943  040530  005705                        TST     R5              ;SEE IF SHOULD INDENT
8944  040532  001002                        BNE     11$             ;BR IF NOT
8945  040534  104401  012427                TYPE    ,SPACE2         ;INDENT
8946  040540  013146                11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
8947  040542  104402                        TYPOC                   ;TYPE IT
8948  040544  005303                        DEC     R3              ;MORE DATA WORDS
8949  040546  001403                        BEQ     12$             ;NO-BRANCH
8950  040550  104401  012427                TYPE    ,SPACE2         ;TYPE SEPARATORS
8951  040554  000771                        BR      11$             ;LOOP
8952  040556  005702                12$:    TST     R2              ;SEE IF <CR>,<LF> NEEDED
8953  040560  001402                        BEQ     14$             ;BR IF NOT
8954  040562  104401  001315                TYPE    ,$CRLF          ;TYPE IT
8955  040566  005302                14$:    DEC     R2              ;MORE DH'S?
8956  040570  003425                        BLE     20$             ;NO-BRANCH
8957  040572  012037  040624        15$:    MOV     (R0)+,16$       ;GET NEXT DH POINTER
8958  040576  105710                        TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
8959  040600  001004                        BNE     34$             ;BR IF YES
8960  040602  104401  001315                TYPE    ,$CRLF          ;SKIP EXTRA LINE
8961  040606  005005                        CLR     R5              ;RE-INIT INDENT INDICATOR
8962  040610  000404                        BR      36$
8963  040612  005105                34$:    COM     R5              ;COMPLEMENT INDENT INDICATOR
8964  040614  001002                        BNE     36$             ;BR IF NO INDENT REQUIRED
8965  040616  104401  012427                TYPE    ,SPACE2         ;INDENT
8966  040622  104401                36$:    TYPE                    ;TYPE DH
8967  040624  000000                16$:    .WORD   0               ;DH POINTER GOES HERE
8968  040626  104401  001315                TYPE    ,$CRLF
8969  040632  105710                        TSTB    (R0)            ;TYPE A DT?
8970  040634  001331                        BNE     10$             ;YES-BRANCH
8971  040636  062700  000002                ADD     #2,R0           ;INCREMENT DF POINTER
8972  040642  000751                        BR      14$             ;SEE IF END OF DF BLOCK
8973  040644  104410                20$:    RESREG
8974  040646  000207                        RTS     PC
8975                                ;;*****************************************************************
8976                                .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
8977                                ;*ENTRY:        JSR PC, CONERR
8978                                ;*RETURN:       RTS PC
8979                                ;*
8980                                ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
8981                                ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
8982                                ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
8983                                ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
8984                                ;*AT THIS TIME.
8985                                ;;*****************************************************************
8986  040650  104407                CONERR: SAVREG                  ;SAVE R0-R5
8987  040652  152737  000377  003113        BISB    #377,DONE       ;SET DONE FLAG
8988  040660  105237  003116                INCB    ERRCNT          ;INCREMENT ERROR COUNT
8989  040664  004737  043600                JSR     PC,TOPROC       ;LOAD RK REGS INTO $REGS
8990  040670  032737  000001  003042        BIT     #BIT0,E.CONT    ;ERROR 0?
8991  040676  001402                        BEQ     1$              ;NO-BRANCH
8992  040700  104064                        ERROR   64              ;CLEAR CONT DID NOT CLEAR ERROR
8993  040702  000470                        BR      7$
8994  040704  032737  000002  003042  1$:   BIT     #BIT1,E.CONT    ;ERROR 1?
```

```
8995  040712  001402                          BEQ     2$              ;NO-BRANCH
8996  040714  104065                          ERROR   65              ;NO ATTENTION IN ATTENTION SJM REG
8997  040716  000462                          BR      7$
8998  040720  032737  000004  003042  2$:     BIT     #BIT2,E.CONT    ;ERROR 2?
8999  040726  001407                          BEQ     3$              ;NO-BRANCH
9000  040730  105737  003127                  TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
9001  040734  001402                          BEQ     15$             ;BR IF NOT
9002  040736  105237  003130                  INCB    REISSU          ;SET FLAG TO RE-ISSUE COMMAND
9003  040742  104066                  15$:    ERROR   66              ;UNSOLICITED ATTENTION
9004  040744  000447                          BR      7$
9005  040746  032737  000010  003042  3$:     BIT     #BIT3,E.CONT    ;ERROR 3?
9006  040754  001402                          BEQ     4$              ;NO-BRANCH
9007  040756  104067                          ERROR   67              ;UNEXPECTED DATA TYPE ERROR
9008  040760  000441                          BR      7$
9009  040762  032737  000020  003042  4$:     BIT     #BIT4,E.CONT    ;ERROR 4?
9010  040770  001402                          BEQ     5$              ;NO-BRANCH
9011  040772  104070                          ERROR   70              ;ATTENTION DID NOT RESET WITH CLEAR
9012  040774  000433                          BR      7$
9013  040776  032737  000040  003042  5$:     BIT     #BIT5,E.CONT    ;ERROR 5?
9014  041004  001402                          BEQ     6$              ;NO-BRANCH
9015  041006  104071                          ERROR   71              ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
9016  041010  000425                          BR      7$
9017  041012  032737  000400  003042  6$:     BIT     #BIT8,E.CONT
9018  041020  001401                          BEQ     8$
9019  041022  104072                          ERROR   72              ;DATA LATE WHEN UNLOADING HEADER
9020  041024  032737  001000  003042  8$:     BIT     #BIT9,E.CONT
9021  041032  001401                          BEQ     9$
9022  041034  104073                          ERROR   73              ;CONTROLLER ERROR DURING DRIVER SERVICE
9023  041036  032737  002000  003042  9$:     BIT     #BIT10,E.CONT
9024  041044  001401                          BEQ     10$
9025  041046  104074                          ERROR   74              ;DRIVE DETECTED PARITY ERROR
9026  041050  032737  100000  003042  10$:    BIT     #BIT15,E.CONT
9027  041056  001401                          BEQ     11$
9028  041060  104052                          ERROR   52              ;MULTIPLE DRIVE SELECT
9029  041062  104075                  11$:    ERROR   75              ;UNDEFINED ERROR
9030  041064  005037  003042          7$:     CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
9031  041070  000137  043400                  JMP     BGNRTY          ;GO DO RETRY
9032                                  ;;*****************************************************************
9033                                  .SBTTL  REPORT SUPPORT ROUTINE
9034                                  ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
9035                                  ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING.  ALL THESE MAY
9036                                  ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
9037  041074                          REPSUP:
9038  041074  104407                          SAVREG
9039  041076  005037  005476                  CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
9040  041102  116537  000001  005476          MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
9041  041110  012700  001162                  MOV     #$REG0,R0       ;FOR REPORTING
9042  041114  016520  000002                  MOV     P.CYLN(R5),(R0)+
9043  041120  116520  000005                  MOVB    P.TRCK(R5),(R0)+
9044  041124  105020                          CLRB    (R0)+
9045  041126  116520  000004                  MOVB    P.SECT(R5),(R0)+
9046  041132  105020                          CLRB    (R0)+
9047  041134  016520  000012                  MOV     P.WC(R5),(R0)+
9048  041140  012700  001174                  MOV     #$REG5,R0
9049  041144  116503  000007                  MOVB    P.BAHI(R5),R3
9050  041150  042703  177774                  BIC     #177774,R3
```

```
 9051  041154  010337  001256                MOV     R3,$REG36       ;HI BA BITS
 9052  041160  016537  000010  001260        MOV     P.BALO(R5),$REG37 ;LO BA BITS
 9053  041166  016520  000016                MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
 9054  041172  016520  000020                MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
 9055  041176  016520  000030                MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
 9056  041202  016520  000026                MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
 9057  041206  016520  000022                MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
 9058                                                                ;FOR ALL REPORTS (TO BE
 9059  041212  016520  000024                MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
 9060  041216  016520  000032                MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
 9061  041222  016520  000036                MOV     P.DS(R5),(R0)+
 9062  041226  016520  000034                MOV     P.ER(R5),(R0)+
 9063  041232  016520  000040                MOV     P.AOO(R5),(R0)+
 9064  041236  016520  000042                MOV     P.BOO(R5),(R0)+
 9065  041242  016520  000044                MOV     P.A01(R5),(R0)+
 9066  041246  016520  000046                MOV     P.B01(R5),(R0)+
 9067  041252  016520  000050                MOV     P.A10(R5),(R0)+
 9068  041256  016520  000052                MOV     P.B10(R5),(R0)+
 9069  041262  016520  000054                MOV     P.A11(R5),(R0)+
 9070  041266  016520  000056                MOV     P.B11(R5),(R0)+
 9071                          ;STORE PREVIOUS COMMAND FOR PRINTOUT
 9072  041272  012701  001236                MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
 9073  041276  012700  005222                MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
 9074  041302  112021                        MOVB    (R0)+,(R1)+     ;DRIVE NO.
 9075  041304  105021                        CLRB    (R1)+
 9076  041306  112021                        MOVB    (R0)+,(R1)+     ;COMMAND
 9077  041310  105021                        CLRB    (R1)+
 9078  041312  012021                        MOV     (R0)+,(R1)+     ;CYL ADDRESS
 9079  041314  116021  000001                MOVB    1(R0),(R1)+     ;TRACK
 9080  041320  105021                        CLRB    (R1)+
 9081  041322  111021                        MOVB    (R0),(R1)+      ;SECTOR
 9082  041324  105021                        CLRB    (R1)+
 9083  041326  016021  000006                MOV     6(R0),(R1)+     ;WORD COUNT
 9084  041332  116003  000003                MOVB    3(R0),R3        ;HI BA BITS
 9085  041336  042703  177774                BIC     #177774,R3
 9086  041342  010321                        MOV     R3,(R1)+
 9087  041344  016011  000004                MOV     4(R0),(R1)      ;LO BA BITS
 9088  041350  104410                        RESREG
 9089  041352  000207                        RTS     PC
 9090                          ;;*****************************************************************
 9091                          .SBTTL  REPORT ERROR ROUTINE
 9092                          ;* ENTRY         JSR     PC, ERRHDL
 9093                          ;*RETURN        RTS     PC
 9094                          ;*
 9095                          ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN AN ERROR IS DETECTED
 9096                          ;*IN THE OPERATION.  THE ROUTINE DETERMINES WHICH COMMAND WAS
 9097                          ;*BEING EXECUTED AND GENERATES THE PROPER ERROR MESSAGE.
 9098                          ;;*****************************************************************
 9099
 9100  041354  104407          ERRHDL: SAVREG
 9101  041356  152737  000377  003113        BISB    #377,DONE       ;SET DONE FLAG
 9102  041364  105237  003116                INCB    ERRCNT          ;INCREMENT ERROR COUNT
 9103  041370  005037  005474                CLR     RECODE          ;CLEAR RECOVERY CODE WORD
 9104  041374  032737  000400  005474  ER2ENT: BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
 9105  041402  001402                        BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
 9106  041404  012705  002704                MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
```

```
                041410  012737  043552  000036  52$:    MOV     #RETANL,@#ABNL  ;SET NOW ABNORMAL AND NORMAL RETURN FOR
                041416  012737  043570  000034          MOV     #RETNML,@#NORM  ;DRIVER OPERATIONS IN ERROR PROCESSING
                041424  004737  041074                  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
                041430  016504  000034                  MOV     P.ER(R5),R4     ;SET R4 TO ERROR REGISTER
                041434  032765  020000  000020          BIT     #UPE,P.CS2(R5)  ;TEST UPE. IF UES-SET
                041442  001406                          BEQ     1$              ;REPORT ERROR
                041444  104001                          ERROR   1
                041446  052737  000200  005474          BIS     #ABORT,RECODE
                041454  000137  043124                  JMP     37$
                041460  032765  004000  000020  1$:     BIT     #NEM,P.CS2(R5)  ;TEST NON-EXISTANT MEMORY
                041466  001406                          BEQ     2$
                041470  104002                          ERROR   2
                041472  052737  000200  005474          BIS     #ABORT,RECODE
                041500  000137  043124                  JMP     37$
                041504  032765  010000  000020  2$:     BIT     #NED,P.CS2(R5)  ;TEST NON-EXISTANT DRIVE
                041512  001412                          BEQ     3$
                041514  032765  000400  000020          BIT     #UFE,P.CS2(R5)  ;TEST IF NED & UFE BOTH SET
                041522  001403                          BEQ     38$
                041524  104035                          ERROR   35              ;NED & UFE BOTH SET ERROR
                041526  000137  043124                  JMP     37$
                041532  104003                  38$:    ERROR   3               ;NED ONLY
                041534  000137  043124                  JMP     37$
                041540  032765  000400  000020  3$:     BIT     #UFE,P.CS2(R5)  ;TEST UNIT FIELD ERROR
                041546  001412                          BEQ     4$
                041550  032765  010000  000020          BIT     #NED,P.CS2(R5)  ;TEST IF UFE & NED BOTH SET
                041556  001403                          BEQ     39$             ;NO-SKIP
                041560  104035                          ERROR   35              ;REPORT NED & UFE BOTH SET
                041562  000137  043124                  JMP     37$
                041566  104004                  39$:    ERROR   4               ;REPORT UFE ONLY
                041570  000137  043124                  JMP     37$
                041574  032765  000100  000014  4$:     BIT     #CMDTO,P.PRST(R5) ;
                041602  001423                          BEQ     5$
                041604  004737  043600                  JSR     PC,TOPROC       ;GO PROCESS TIMEOUT
                041610  032737  002000  005474          BIT     #TWOTOS,RECODE  ;2ND TIMEOUT IN TIMEOUT PROC?
                041616  001007                          BNE     40$             ;YES - SKIP TO ERROR 56
                041620  032737  000400  005474          BIT     #LEV2ER,RECODE  ;TEST IF LEVEL 2 ERROR
                041626  001006                          BNE     41$             ;YES - SKIP TO ERROR 57
                041630  104005                          ERROR   5               ;ELSE MAKE FULL TIMEOUT REPORT
                041632  000137  043124                  JMP     37$
```

C14


```
041636  104056                          40$:    ERROR   56                      ;TWO TIMEOUTS ERROR REPORT
041640  000137  043124                          JMP     37$
041644  104057                          41$:    ERROR   57                      ;2ND LEVEL ERROR REPORT
041646  000137  041374                          JMP     ER2ENT                  ;GO BUILD AND MAKE 2ND REPORT
041652  032765  010000  000014          5$:     BIT     #DRVSZD,P.PRST(R5)      ;SEE IF DRIVE SIEZED BY OTHER PORT
041660  001431                                  BEQ     65$                     ;BR IF DRIVE IS AVAILABLE
041662  105737  003126                          TSTB    DULACS                  ;SEE IF DUAL-ACCESS FLAG SET
041666  001003                                  BNE     63$                     ;BR IF DUAL-ACCESS TEST
041670  104107                                  ERROR   107                     ;DRIVE SIEZED BY OTHER PORT
041672  000137  043124                          JMP     37$
041676  105237  003127                  63$:    INCB    DRNAFG                  ;SET DRIVE NOT AVAILABLE FLAG
041702  012762  100000  000000                  MOV     #100000,RKCS1(R2)       ;CLEAR CONTROLLER ERROR
041710  013737  003056  003104                  MOV     W.MIN,W.DRV             ;SET TIMER FOR ABOUT A MINUTE
041716  113700  005500                          MOVB    DRIVE,R0                ;DRIVE NUMBER
041722  116037  003072  003071                  MOVB    I.DRV(R0),INTMSK        ;SET MASK FOR THIS DRIVE
041730  113737  003071  003070                  MOVB    INTMSK,W.TIME           ;SET DRIVE NUMBER FOR THIS DRIVE
041736  105037  003113                          CLRB    DONE                    ;CLEAR THE DONE FLAG
041742  000207                                  RTS     PC                      ;GO WAIT FOR DRIVE ATT'N
041744  032765  020000  000016          65$:    BIT     #SPAR,P.CS1(R5)         ;TEST D TO C PARITY ERROR
041752  001406                                  BEQ     6$
041754  104006                                  ERROR   6
041756  052737  004000  005474                  BIS     #RCLREQ,RECODE
041764  000137  043124                          JMP     37$
041770  032704  000010                  6$:     BIT     #DRPAR,R4               ;TEST DRIVE DETECTED PARITY ERROR
041774  001406                                  BEQ     7$
041776  104007                                  ERROR   7
042000  052737  004000  005474                  BIS     #RCLREQ,RECODE
042006  000137  043124                          JMP     37$
042012  032765  000010  000036          7$:     BIT     #ACLO,P.DS(R5)          ;TEST AC LOW
042020  001403                                  BEQ     8$
042022  104010                                  ERROR   10
042024  000137  043124                          JMP     37$
042030  032765  000020  000036          8$:     BIT     #SPDLSS,P.DS(R5)        ;TEST SPEED LOSS
042036  001403                                  BEQ     24$
042040  104011                                  ERROR   11
042042  000137  043124                          JMP     37$
042046  032765  004000  000016          24$:    BIT     #CTO,P.CS1(R5)          ;TEST FOR CONTROLLER TIMEOUT
042054  001401                                  BEQ     25$
042056  104027                                  ERROR   27
042060  032704  000001                  25$:    BIT     #ILC,R4                 ;TEST ILLEGAL FUNCTION CODE
042064  001403                                  BEQ     10$
042066  104012                                  ERROR   12
042070  000137  043124                          JMP     37$
042074  032765  002000  000020          10$:    BIT     #PGE,P.CS2(R5)          ;TEST PROGRAMMING ERROR
042102  001403                                  BEQ     11$
042104  104013                                  ERROR   13
042106  000137  043124                          JMP     37$
042112  032704  000004                  11$:    BIT     #ILF,R4                 ;TEST ILLEGAL DRIVE FUNCTION
042116  001403                                  BEQ     12$
042120  104014                                  ERROR   14
042122  000137  043124                          JMP     37$
042126  032704  000040                  12$:    BIT     #DTYE,R4                ;TEST DRIVE TYPE ERROR
042132  001403                                  BEQ     13$
042134  104015                                  ERROR   15
042136  000137  043124                          JMP     37$
042142  032704  000020                  13$:    BIT     #FMTE,R4                ;TEST FORMAT ERROR
```

```
                                                BEQ     14$
                                                ERROR   16
                                                JMP     37$
                                        14$:    BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
                                                BEQ     15$
                                                ERROR   17
                                                JMP     37$
                                        15$:    BIT     #UNS,R4         ;TEST DRIVE UNSAFE
                                                BEQ     16$
                                                ERROR   20
                                                BIS     #ABORT,RECODE
                                                JMP     ALLTRM
                                        16$:    BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
                                                BEQ     17$
                                                ERROR   21
                                                BIS     #RCLREQ,RECODE
                                                JMP     37$
                                        17$:    BIT     #COE,R4         ;TEST CYLINDER OVERFLOW
                                                BEQ     18$
                                                ERROR   22
                                                BIS     #RCLREQ,RECODE
                                                JMP     37$
                                        18$:    BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
                                                BEQ     19$
                                                ERROR   23
                                                BIS     #RCLREQ,RECODE
                                                JMP     37$
                                        19$:    BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
                                                BEQ     20$
                                                ERROR   24
                                                BIS     #RCLREQ,RECODE
                                                JMP     37$
                                        20$:    BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
                                                BEQ     21$
                                                ERROR   25
                                                BIS     #ABORT,RECODE
                                                JMP     37$
                                        21$:    BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
                                                BEQ     22$
                                                ERROR   26
                                                JMP     37$
                                        22$:    BIT     #OPI,R4         ;TEST IF OPI ERROR
                                                BEQ     29$
                                                BIS     #OPIERR,RECODE
                                                TSTB    DERCNT          ;TEST IF FIRST ERROR
                                                BEQ     50$
                                                JMP     37$             ;NO - SKIP REPORT
                                        50$:    BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
                                                BEQ     26$             ;HAS ALREADY OCCURRED
                                        27$:    ERROR   54
                                                JMP     37$             ;GET OUT OF ERROR REPORT
                                        26$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
                                                JSR     PC,RDHDC        ;GET HEADER OF SECTOR 0
                                                BIT     #LEV2ER,RECODE  ;TEST IF ERROR GETTING HDR
                                                BNE     28$             ;IF YES-GO MAKE ABREVIATED REPORT
                                                MOV     RKBAS,R2        ;STORE HEADER 0 INTO REGISTERS
```

```
9275  042456  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;RESET INTERRUPT ENABLE
9276  042464  016237  000024  001202          MOV     RKDB(R2),$REG10 ;FOR REPORTING
9277  042472  016237  000024  001204          MOV     RKDB(R2),$REG11
9278  042500  016237  000024  001206          MOV     RKDB(R2),$REG12
9279  042506  032762  100000  000000          BIT     #CERR,RKCS1(R2) ;TEST IF ERROR DURING STORAGE
9280  042514  001343                          BNE     27$
9281  042516  104030                          ERROR   30              ;MAKE OPI REPORT
9282  042520  000137  043124                  JMP     37$
9283  042524  104054                  28$:    ERROR   54
9284  042526  000137  041374                  JMP     ER2ENT          ;GO MAKE 2ND LEVEL REPORT
9285  042532  032704  000400          29$:    BIT     #HVRC,R4        ;TEST IF HVRC ERROR
9286  042536  001457                          BEQ     23$
9287  042540  052737  000004  005474          BIS     #HVRCER,RECODE
9288  042546  105737  003111                  TSTB    DERCNT          ;TEST IF FIRST ERROR
9289  042552  001402                          BEQ     30$             ;YES - REPORT
9290  042554  000137  043124                  JMP     37$             ;JUMP TO RETURN
9291  042560  032737  000400  005474  30$:    BIT     #LEV2ER,RECODE  ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
9292  042566  001403                          BEQ     31$             ;OCCURRED. NO-SKIP EXIT
9293  042570  104055                  51$:    ERROR   55
9294  042572  000137  043124                  JMP     37$             ;GET OUT OF ERROR REPORT
9295  042576  004737  044230          31$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
9296  042602  004737  043750                  JSR     PC,RDHDO        ;GO GET HDR 0
9297  042606  032737  000400  005474          BIT     #LEV2ER,RECODE  ;TEST IF ERROR IN GETTING HDR
9298  042614  001025                          BNE     32$             ;IF YES-GO MAKE ABREVIATED REPORT
9299  042616  013702  003026                  MOV     RKBAS,R2        ;GET RK611 BASE ADDRESS
9300  042622  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;CLEAR INTERRUPT ENABLE
9301  042630  016237  000024  001202          MOV     RKDB(R2),$REG10 ;STORE OFF HEADER
9302  042636  016237  000024  001204          MOV     RKDB(R2),$REG11
9303  042644  016237  000024  001206          MOV     RKDB(R2),$REG12
9304  042652  032762  100000  000000          BIT     #CERR,RKCS1(R2) ;TEST IN ANY ERROR IN UNLOAD
9305  042660  001343                          BNE     51$             ;IY YES - GO MAKE SHORT REPORT
9306  042662  104031                          ERROR   31              ;MAKE FULL REPORT
9307  042664  000137  043124                  JMP     37$
9308  042670  104055                  32$:    ERROR   55              ;ABREVIATED HVRC ERROR RPORT
9309  042672  000137  041374                  JMP     ER2ENT          ;GO REPORT 2ND LEVEL ERROR
9310  042676  032704  000200          23$:    BIT     #BSE,R4         ;TEST FOR BAD SECTOR ERROR
9311  042702  001430                          BEQ     33$             ;NO - SKIP
9312  042704  052737  000002  005474          BIS     #BSERR,RECODE   ;SET ERROR FLAG
9313  042712  016500  000030                  MOV     P.DCYL(R5),R0   ;GET CYL NO.
9314  042716  116501  000027                  MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
9315  042722  042701  177774                  BIC     #177774,R1      ;CLEAR ALL BITS EXCEPT TRACK
9316  042726  016502  000026                  MOV     P.DTS(R5),R2    ;GET SECTOR IN ERROR
9317  042732  042702  177740                  BIC     #177740,R2      ;CLEAR ALL BITS EXCEPT SECTOR
9318  042736  004737  037556                  JSR     PC,BDSRCK       ;GO SEE IF THIS SECTOR LISTED
9319  042742  032737  001000  005474          BIT     #BADSEC,RECODE  ;TEST RESULT
9320  042750  001065                          BNE     37$             ;YES - EXIT, NO ERROR OR REPORT
9321  042752  104104                          ERROR   104             ;ELSE REPORT BAD BSE
9322  042754  042737  000002  005474          BIC     #BSERR,RECODE   ;RESET BSE ERROR FLAG
9323  042762  000460                          BR      37$             ;GO EXIT
9324  042764  032704  100000          33$:    BIT     #DCK,R4         ;TEST IF DATA CHECK
9325  042770  001435                          BEQ     36$
9326  042772  052737  000020  005474          BIS     #DCKERR,RECODE  ;SET DATA CHECK ERROR IN RECOVERY CODE
9327  043000  032704  000100                  BIT     #ECH,R4         ;TEST IF ECC IS HARD. IF
9328  043004  001406                          BEQ     34$             ;YES SET UNCORRECTABLE IN
9329  043006  052737  000040  005474          BIS     #ECCNC,RECODE   ;RECOVERY FLAG AND A 0 IN
9330  043014  005037  001206                  CLR     $REG12          ;REG12 TO INDICATE UNCORRECTABLE.
```

```
9331  043020  000403                          BR      35$             ;A 1 IN REG 12 FOR CORRECTABLE
9332  043022  012737  000001  001206  34$:    MOV     #1,$REG12
9333  043030  105737  003111          35$:    TSTB    DERCNT          ;TEST IF FIRST ERROR
9334  043034  001033                          BNE     37$             ;NO SKIP REPORT
9335  043036  004737  044134                  JSR     PC,GTPKAD       ;GO GET PACK ADDRESS OF ERROR
9336  043042  016537  000060  001202          MOV     P.EPOS(R5),$REG10 ;STORE ECC POSITION &
9337  043050  016537  000062  001204          MOV     P.EPAT(R5),$REG11 ;PATTERN
9338  043056  104032                          ERROR   32              ;REPORT DCK ERROR
9339  043060  000137  043124                  JMP     37$
9340  043064  032765  040000  000020  36$:    BIT     #WCE,P.CS2(R5)  ;TEST WRITE CHECK ERROR
9341  043072  001414                          BEQ     37$
9342  043074  042737  000200  005474          BIC     #ABORT,RECODE   ;CLEAR ABORT & SET WRITE
9343  043102  052737  000100  005474          BIS     #WCERR,RECODE   ;CHECK ERROR IN RECODE
9344  043110  105737  003111                  TSTB    DERCNT          ;TEST IF FIRST ERROR
9345  043114  001003                          BNE     37$             ;NO - SKIP
9346  043116  004737  044104                  JSR     PC,GTPKAD       ;GO GET ADDRESS OF ERROR
9347  043122  104033                          ERROR   33              ;REPORT WCE
9348
9349  043124  032765  000020  000014  37$:    BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
9350  043132  001404                          BEQ     43$
9351  043134  104036                          ERROR   36
9352  043136  052737  000200  005474          BIS     #ABORT,RECODE
9353
9354  043144  032765  000040  000014  43$:    BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
9355  043152  001404                          BEQ     44$
9356  043154  104037                          ERROR   37
9357  043156  052737  000200  005474          BIS     #ABORT,RECODE
9358
9359  043164  032765  004000  000014  44$:    BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
9360  043172  001404                          BEQ     46$
9361  043174  104040                          ERROR   40
9362  043176  052737  000200  005474          BIS     #ABORT,RECODE
9363
9364  043204  032765  000010  000014  46$:    BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
9365  043212  001404                          BEQ     ALLTRM
9366  043214  104042                          ERROR   42
9367  043216  052737  000200  005474          BIS     #ABORT,RECODE
9368
9369
9370                                  ;ALL ERRORS MUST EXIT THROUGH THIS POINT
9371
9372  043224  012705  002620          ALLTRM: MOV     #PARMO,R5       ;RESTORE PARAMETER BLOCK SELECTION
9373  043230  012737  041354  003036          MOV     #ERRHDL,A.ABNL  ;RESTORE INTERRUPT VECTORS FOR RETRY
9374  043236  012737  040106  003034          MOV     #ERRFRE,A.NORM  ;DRIVER OPERATIONS, IF ANY
9375  043244  032737  000200  005474          BIT     #ABORT,RECODE   ;IF ABORT IS NOT SET AND
9376  043252  001043                          BNE     48$             ;THE DRIVE IS READY (HAS NOT
9377                                                                  ;CYCLED DOWN)
9378  043254  013702  003026                  MOV     RKBAS,R2        ;GET BASE ADDRESS
9379  043260  032762  000200  000012          BIT     #RDY,RKDS(R2)   ;TEST IF DRIVE READY SET
9380  043266  001004                          BNE     47$             ;RECALIBRATE REQUIRED BIT IS SET
9381  043270  052737  000200  005474          BIS     #ABORT,RECODE   ;ELSE ABORT WITH ABORT MESSAGE
9382  043276  000431                          BR      48$
9383  043300  032737  004000  005474  47$:    BIT     #RCLREQ,RECODE  ;IF RECALIBRATE IS REQUIRED
9384  043306  001434                          BEQ     BGNRTY          ;FOR RETRY SET UP PARAM
9385  043310  112737  000113  000001          MOVB    #RECAL,P.CMND   ;BLOCK TO DO IT.
9386  043316  012737  043552  003036          MOV     #RETANL,A.ABNL
```

```
                        REPORT ERROR ROUTINE

 9387  043324  004737  037662                    JSR     PC,DRVCAL
 9388  043330  012737  041354  003036            MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR RETURN
 9389  043336  032737  000400  005474            BIT     #LEV2ER,RECODE  ;IF AN ERROR OCCURRED IN THE
 9390  043344  001415                            BEQ     BGNRTY          ;RECAL ATTEMP SET ABORT,
 9391  043346  052737  000200  005474            BIS     #ABORT,RECODE   ;PRINT THE RECAL ERROR MESSAGE, AND
 9392  043354  104060                            ERROR   60              ;GO REPORT DETAILS
 9393  043356  000137  041374                    JMP     ER2ENT
 9394  043362  104061                    48$:    ERROR   61              ;REPORT ABORT-RETRY FAILED
 9395
 9396                                    ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
 9397                                    ;IF THE DRIVE READY BIT IS RESET.
 9398
 9399  043364  000000                    HLTPRG: HALT
 9400  043366  105037  003116                    CLRB    ERRCNT          ;CLEAR ERROR COUNT
 9401  043372  000005                            RESET                   ;RESET ALL DEVICES
 9402  043374  000137  012542                    JMP     CMSTRT
 9403
 9404
 9405                                    ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
 9406                                    ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
 9407                                    ; THE INITIATING ROUTINE FOR RETRY.  ANY OTHER ERROR IS TO BE
 9408                                    ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
 9409                                    ;FLAG IS SET AND PROGRAM HALTS.
 9410
 9411  043400  032737  000136  005474    BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
 9412  043406  001404                            BEQ     3$
 9413  043410  052737  100000  005474    9$:     BIS     #ANYDER,RECODE
 9414  043416  000453                            BR      2$
 9415  043420                            3$:
 9416  043420  105737  003133                    TSTB    NORTRY          ;SEE IF "NO-RETRY" FLAG SET
 9417  043424  001371                            BNE     9$              ;BR IF YES
 9418  043426  032737  001000  005474            BIT     #BADSEC,RECODE  ;TEST IF BAD SECTOR FLAG SET
 9419  043434  001044                            BNE     2$              ;IF YES-EXIT TO CALLER
 9420  043436  123737  003116  003117            CMPB    ERRCNT,ERRLMT   ;BEGIN RETRY IF ERROR COUNT HAS
 9421  043444  001012                            BNE     1$              ;NOT BEEN EXCEEDED
 9422  043446  005037  001174                    CLR     $REGS
 9423  043452  113737  003116  001174            MOVB    ERRCNT,$REGS    ;GET ERROR RETRY COUNT
 9424  043460  104102                            ERROR   102             ;REPORT RETRY UNSUCCESSFUL
 9425  043462  052737  000200  005474            BIS     #ABORT,RECODE   ;SET ABORT & QUIT
 9426  043470  000735                            BR      HLTPRG
 9427  043472  013702  003026            1$:     MOV     RKBAS,R2        ;GET RK BASE ADDRESS
 9428  043476  112765  000177  000000            MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
 9429  043504  004737  037662                    JSR     PC,DRVCAL       ;GO DO IT
 9430  043510  012700  005236                    MOV     #COMSTR,R0      ;GO AND REESTABLISH THE COMMAND
 9431  043514  012025                            MOV     (R0)+,(R5)+     ;AS IT WAS ENTERED INTO THE
 9432  043516  012025                            MOV     (R0)+,(R5)+     ;PARAMETER BLOCK
 9433  043520  012025                            MOV     (R0)+,(R5)+
 9434  043522  012025                            MOV     (R0)+,(R5)+
 9435  043524  012025                            MOV     (R0)+,(R5)+
 9436  043526  012025                            MOV     (R0)+,(R5)+
 9437  043530  012705  002620                    MOV     #PARM0,R5
 9438  043534  012737  000000  177776            MOV     #PR0,@#PSW      ;LOWER PRIORITY TO ALLOW INTERRUPT
 9439  043542  004737  037662                    JSR     PC,DRVCAL       ;CALL DRIVER
 9440  043546  104410                    2$:     RESREG                  ;IF RETURN GETS HERE, NO ERROR
 9441  043550  000207                            RTS     PC              ;OCCURRED, RECOVERY WAS SUCCESSFUL
 9442
```

```
043552  152737  000377  003113  RETANL: BISB    #377,DONE          ;SET DONE
043560  052737  000400  005474          BIS     #LEV2ER,RECODE     ;SET LEVEL TWO ERROR
043566  000207                          RTS     PC
043570  152737  000377  003113  RETNML: BISB    #377,DONE          ;SET DONE
043576  000207                          RTS     PC



                                ;;*******************************************************************
                                .SBTTL  TIME OUT PROCESSOR ROUTINE
                                ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
                                ;*GATHERING DUTIES.
                                ;;*******************************************************************
043600                          TOPROC:
043600  104407                          SAVREG
043602  013702  003026                  MOV     RKBAS,R2
043606  012701  001174                  MOV     #$REGS,R1          ;SET UP R1 FOR RK REGISTER STORAGE
043612  016221  000000                  MOV     RKCS1(R2),(R1)+    ;STORE ALL VALID RK611
043616  016221  000010                  MOV     RKCS2(R2),(R1)+    ;REGISTERS
043622  016221  000020                  MOV     RKDC(R2),(R1)+
043626  016221  000006                  MOV     RKDA(R2),(R1)+
043632  016221  000002                  MOV     RKWC(R2),(R1)+
043636  016221  000004                  MOV     RKBA(R2),(R1)+
043642  016221  000016                  MOV     RKASOF(R2),(R1)+
043646  016221  000012                  MOV     RKDS(R2),(R1)+
043652  016221  000014                  MOV     RKER(R2),(R1)+
043656  005000                          CLR     R0                 ;THIS CODE WILL ATTEMPT TO
                                                                   ;RETRIEVE THE STATUS FROM THE
                                                                   ;DRIVE.
043660  012705  002704                  MOV     #PARM1,R5          ;SET UP TO USE PARM1
043664  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
043672  004737  037662                  JSR     PC,DRVCAL          ;CALL DRIVER
043676  032765  000100  000014          BIT     #CMDTO,P.PRST(R5)  ;TEST FOR TIMEOUT
043704  001403                          BEQ     1$                 ;NO - SKIP
043706  052737  002000  005474          BIS     #TWOTOS,RECODE     ;SET TWO TIMEOUTS FLAG
043714  062705  000040          1$:     ADD     #P.ADD,R5          ;BUMP R5 TO POINT TO DRIVE STATUS
043720  012521                          MOV     (R5)+,(R1)+        ;MOVE ALL THE DRIVE STATUS INTO THE
043722  012521                          MOV     (R5)+,(R1)+        ;TEMP REGS FOR REPORTING.
043724  012521                          MOV     (R5)+,(R1)+
043726  012521                          MOV     (R5)+,(R1)+
043730  012521                          MOV     (R5)+,(R1)+
043732  012521                          MOV     (R5)+,(R1)+
043734  012521                          MOV     (R5)+,(R1)+
043736  012521                          MOV     (R5)+,(R1)+

043740  012705  002620                  MOV     #PARM0,R5          ;RESTORE PARM 0
043744  104410                          RESREG
043746  000207                          RTS     PC



                                ;;*******************************************************************
                                .SBTTL  READ HEADER 0 ROUTINE
                                ;;*******************************************************************
043750                          RDHD0:
043750  104407                          SAVREG
```

```
 9499  043752  016501  000026                    MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
 9500  043756  016500  000052                    MOV     P.BIO(R5),R0    ;GET THE CYLINDER ADRS
 9501  043762  042700  160017                    BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
 9502  043766  006200                            ASR     R0              ;OFF UNUSED BITS AND POSITION
 9503  043770  006200                            ASR     R0              ;FOR USE AS THE DESIRED
 9504  043772  006200                            ASR     R0              ;CYLINDER IN THE READ
 9505  043774  006200                            ASR     R0              ;HEADER COMMAND.
 9506  043776  012705  002704                    MOV     #PARM1,R5       ;SET UP TO USE P.B. 1
 9507  044002  010165  000004                    MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
 9508  044006  010065  000002                    MOV     R0,P.CYLN(R5)   ;SET CYL NO.
 9509  044012  132737  000020  003115            BITB    #B.CFMT,FORMAT  ;TEST PRESENT FMT AND CHANGE IT
 9510  044020  001404                            BEQ     1$              ;TO THE OPPOSITE. THIS WILL CAUSE
 9511  044022  142765  000020  000007            BICB    #B.CFMT,P.CS1H(R5) ;A READ OF SECT 0 HDR ON
 9512  044030  000403                            BR      2$              ;THE READ HDR COMMAND
 9513  044032  152765  000020  000007  1$:       BISB    #B.CFMT,P.CS1H(R5)
 9514  044040  112765  000125  000001  2$:       MOVB    #RDHEAD,P.CMND(R5) ;SET READ HDR COMMAND
 9515  044046  012737  000000  177776            MOV     #PR0,@#PSW      ;ALLOW INTERRUPTS
 9516  044054  004737  037662                    JSR     PC,DRVCAL       ;DO READ HDR
 9517  044060  142765  000020  000007            BICB    #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
 9518  044066  153765  003115  000007            BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
 9519  044074  012705  002620                    MOV     #PARM0,R5       ;RESTORE P.B. 0 ADDRESS
 9520  044100  104410                            RESREG                  ;RESTORE R0-R5
 9521  044102  000207                            RTS     PC              ;RETURN
 9522
 9523
 9524
 9525                                    ;;*****************************************************************
 9526                                    .SBTTL  GET PACK ADDRESS ROUTINE
 9527                                    ;;*****************************************************************
 9528  044104  016537  000030  001174    $TPKAD: MOV     P.DCYL(R5),$REGS ;GET CYLINDER NUMBER
 9529  044112  005037  001176                    CLR     $REG6           ;CLEAR REGISTERS FOR
 9530  044116  005037  001200                    CLR     $REG7           ;TRACK & SECTOR STORAGE
 9531  044122  116537  000026  001200            MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
 9532  044130  116537  000027  001176            MOVB    P.DTS+1(R5),$REG6
 9533  044136  005737  001200                    TST     $REG7           ;ADJUST THE ADDRESS CONTAINED IN
 9534                                                                    ;THE RK REGISTERS FOR THE AUTOMATIC
 9535  044142  001403                            BEQ     1$              ;INCREMENT
 9536  044144  005337  001200                    DEC     $REG7
 9537  044150  000426                            BR      5$
 9538  044152  032765  010000  000016  1$:       BIT     #CFMT,P.CS1(R5)
 9539  044160  001404                            BEQ     2$
 9540  044162  012737  000023  001200            MOV     #19.,$REG7
 9541  044170  000403                            BR      3$
 9542  044172  012737  000025  001200  2$:       MOV     #21.,$REG7
 9543  044200  005737  001176  3$:       TST     $REG6
 9544  044204  001403                            BEQ     4$
 9545  044206  005337  001176                    DEC     $REG6
 9546  044212  000405                            BR      5$
 9547  044214  012737  000002  001176  4$:       MOV     #2,$REG6
 9548  044222  005337  001174                    DEC     $REG5
 9549  044226  000207                    5$:     RTS     PC
 9550
 9551
 9552
 9553                                    ;;*****************************************************************
 9554                                    .SBTTL  BUILD EXPECTED HEADER
```

```
 9555                                   ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
 9556                                   ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
 9557                                   ;*7 FOR REPORTING.
 9558                                   ;;*****************************************************************
 9559  044230  104407                   BLDEXH: SAVREG
 9560  044232  016537  000030  001174           MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
 9561  044240  016501  000026                   MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
 9562  044244  042701  174377                   BIC     #174377,R1      ;CLEAR ALL BUT TRACK BITS
 9563  044250  006201                           ASR     R1              ;AND SECTOR. SHIFT THE TRACK
 9564  044252  006201                           ASR     R1              ;OVER TO CONFORM TO HEADER FORMAT
 9565  044254  006201                           ASR     R1              ;CHECK THE FORMAT BIT AND
 9566                                                                   ;IF SET, SET THE HEADER FORMAT
 9567  044256  016537  000026  001176           MOV     P.DTS(R5),$REG6 ;BIT.
 9568  044264  042737  177740  001176           BIC     #177740,$REG6   ;CLEAR ALL BUT SECTOR
 9569  044272  060137  001176                   ADD     R1,$REG6        ;ADD TRACK AND SECTOR TOGETHER
 9570  044276  052737  140000  001176           BIS     #140000,$REG6   ;INSERT BSE BITS
 9571  044304  032765  010000  000016           BIT     #CFMT,P.CS1(R5)
 9572  044312  001403                           BEQ     23$
 9573  044314  052737  001000  001176           BIS     #1000,$REG6
 9574  044322  013737  001174  001200  23$:     MOV     $REG5,$REG7     ;COMPLTE THE HEADER VRC
 9575  044330  013701  001176                   MOV     $REG6,R1
 9576  044334  043737  001176  001200           BIC     $REG6,$REG7
 9577  044342  043701  001174                   BIC     $REG5,R1
 9578  044346  050137  001200                   BIS     R1,$REG7
 9579  044352  104410                           REGREG
 9580  044354  000207                           RTS     PC
 9581
```

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

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

                                .SBTTL   *WATCH-DOG TIMER

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

044356  010546                  W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
044360  010446                          MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
044362  010346                          MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
044364  010246                          MOV     R2,-(SP)        ;SAVE R2 ON STACK
044366  013746  177776                  MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
044372  005337  003046                  DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
044376  001034                          BNE     20$             ;IF NOT ZERO RETURN
044400  013737  003050  003046          MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
044406  105737  003070                  TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
044412  001426                          BEQ     20$             ;NO, RETURN
044414  013737  003032  177776          MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
044422  013702  003026                  MOV     RKBAS,R2        ;LOAD BASE OF RK06 REGISTERS
044426  005337  003104                  DEC     W.DRV           ;DECREMENT COMMAND TIMER
044432  001016                          BNE     20$             ;RETURN IF NO TIME OUT
```

```
 9638  044434  105037  003070                  CLRB    W.TIME          ;RESET TIMING INDICATOR
 9639  044440  013705  003102                  MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
 9640                                                                  ;  TABLE FOR INDEXING
 9641  044444  052765  000100  000014          BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
 9642  044452  020537  003044                  CMP     R5,0.WAIT       ;CHECK IF DRIVER IS WAITING FOR
 9643                                                                  ;  COMMAND COMPLETION
 9644  044456  001002                          BNE     5$              ;NO, DO NOT ALTER WAITING FOR
 9645                                                                  ;  COMMAND COMPLETION
 9646  044460  005037  003044                  CLR     0.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9647  044464  004737  047740          5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
 9648  044470  012637  177776          20$:    MOV     (SP)+,PS        ;RESTORE PSW
 9649  044474  012602                          MOV     (SP)+,R2        ;RESTORE R2
 9650  044476  012603                          MOV     (SP)+,R3        ;RESTORE R3
 9651  044500  012604                          MOV     (SP)+,R4        ;RESTORE R4
 9652  044502  012605                          MOV     (SP)+,R5        ;RESTORE R5
 9653  044504  000207                          RTS     PC              ;RETURN
```

```
                                .SBTTL   *RK06 INTERRUPT SERVICE ROUTINE

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

044506  010546          I.INTR: MOV     R5,-(SP)        ;STORE R5 ON THE STACK
044510  010446                  MOV     R4,-(SP)        ;STORE R4 ON THE STACK
044512  010346                  MOV     R3,-(SP)        ;STORE R3 ON THE STACK
044514  010246                  MOV     R2,-(SP)        ;STORE R2 ON THE STACK
044516  010146                  MOV     R1,-(SP)        ;STORE R1 ON THE STACK
044520  010046                  MOV     R0,-(SP)        ;STORE R0 ON THE STACK
044522  013702  003026          MOV     RKBAS,R2        ;LOAD R2 TO ADDRESS RK06 REGISTER
044526  016237  000010  002772  MOV     RKCS2(R2),T.CS2 ;STORE CS2
044534  032737  001000  002772  BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
044542  001407                  BEQ     1$              ;NO, CONTINUE PROCESSING
044544  052737  100000  003042  BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
044552  004737  047764          JSR     PC,R.CONT       ;REPORT ERROR
```

```
 9710  044556  000137  046736                JMP     I.RTRN          ;RETURN
 9711
 9712  044562  105737  003064      1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
 9713  044566  001410                        BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
 9714  044570  100403                        BMI     5$              ;CHECK IF RELEASE COMMAND
 9715  044572  105037  003064              CLRB    I.ISRL          ;YES,CLEAR FLAG
 9716  044576  000473                        BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
 9717
 9718  044600  105037  003064      5$:       CLRB    I.ISRL          ;CLEAR FLAG
 9719  044604  000137  045720                JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
 9720
 9721  044610  032737  010400  002772  6$:   BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
 9722                                                                ;  UNIT FIELD ERROR
 9723  044616  001413                        BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
 9724  044620  013704  002772                MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
 9725  044624  042704  177770                BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
 9726  044630  013705  003102                MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
 9727  044634  016237  000000  002770        MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
 9728  044642  000137  045130                JMP     I.ERRC          ;REPORT ERROR
 9729
 9730  044646  016237  000012  003010  7$:   MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
 9731  044654  032737  000001  003010        BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
 9732                                                                ;  PORT
 9733  044662  001041                        BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
 9734
 9735                                        ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
 9736  044664  032737  164000  002772        BIT     #DLT!WCE!UPE!NEM,T.CS2
 9737
 9738  044672  001007                        BNE     10$             ;INDICATE ERROR
 9739  044674  016237  000014  003006        MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
 9740
 9741                              ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
 9742  044702  032737  125700  003006        BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
 9743
 9744  044710  001407                        BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
 9745
 9746  044712  052737  000010  003042  10$:   BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
 9747  044720  004737  047764                JSR     PC,R.CONT       ;REPORT ERROR
 9748  044724  000137  046736                JMP     I.RTRN          ;RESTORE REGISTERS
 9749
 9750  044730  105037  003070      11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 9751  044734  005037  003104                CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
 9752  044740  013705  003102                MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
 9753                                                                ;  ADDRESS
 9754  044744  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
 9755                                                                ;  PROGRAM DRIVE STATUS REGISTER
 9756  044752  005037  003044                CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9757  044756  004737  047740                JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
 9758  044762  000137  046736                JMP     I.RTRN          ;GO RESTORE REGISTERS
 9759
 9760  044766  013705  003044      I.IOO:    MOV     O.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
 9761  044772  001002                        BNE     2$              ;IS COMMAND WAITING PROCESSING
 9762                                                                ;  YES, DO PROCESSING
 9763  044774  000137  045720                JMP     I.ATTN          ;NO, PROCESS ATTENTION
 9764
 9765  045000  013704  002772      2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
                                                BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
                                                
                                                
                                                CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
                                                BEQ     3$              ;YES, CONTINUE
                                                HALT                    ;NO, DRIVER ERROR
                                        000164  CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
                                                BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
                                                JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
                                                
                                        003044  CLR     Q.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
                                        003104  CLR     W.DRV           ;CLEAR WATCH-DOG TIME
                                        003070  CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
                                        002770  MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
                                        002770  BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
                                                BNE     I.ERRC          ;YES, PROCESS ERROR
                                        003004  MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
                                        003005  BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
                                                BNE     15$             ;YES, REPORT ERROR
                                                JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
                                                JMP     I.RTRN          ;RESTORE REGISTERS
                                                
                                        000014  15$:    BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
                                                
                                                I.ERRA: JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
                                                        BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
                                                
                                        000016  I.ERRC: MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
                                                        JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
                                        000062  I.ERR:  MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
                                        000060          MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
                                                        JSR     R0,I.CCLR       ;CLEAR CONTROLLER
                                                        I.RTRN                  ;ERROR RETURN
                                        000020          BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
                                                                                ;  UNIT FIELD ERROR
                                                        BNE     5$              ;YES, REPORT ERROR
                                                        JSR     R0,I.STAT       ;GATHER DRIVE STATUS
                                                        I.RTRN                  ;ERROR RETURN
                                                        MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
                                                        JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
                                                        I.RTRN                  ;ERROR RETURN
                                        003005          BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
                                                        BEQ     2$              ;NO, INDICATE DRIVE ERROR
                                        003042          BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
                                                                                ;  WITH CLEAR
                                                        JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
                                                        JMP     I.RTRN          ;GO RESTORE REGISTERS
                                                
                                        003014  2$:     BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
                                                        BEQ     3$              ;YES, CHECK FAULT
                                                        BIS     #DPVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
                                        003016  3$:     BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
                                                        BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
                                        003042          BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
                                                        JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
                                                        JMP     I.RTRN          ;RETURN
```

```
045310  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
045316  001017                          BNE     10$                ;YES, GO REPORT ERROR
045320  032737  020000  003014          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS CYCLING DOWN
045326  001413                          BEQ     10$                ;NO, REPORT ERROR
045330  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
045336  113737  003071  003070          MOVB    INTMSK,W.TIME      ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
045344  013737  003054  003104          MOV     W.8SEC,W.DRV
045352  000137  046736                  JMP     I.RTRN             ;GO RESTORE REGISTERS

045356  004737  047740          10$:    JSR     PC,R.ABNL          ;GO REPORT ERROR
045362  000137  046736                  JMP     I.RTRN             ;GO RESTORE REGISTERS

                                .SBTTL  *READ ALL HEADERS INTERRUPT SEQUENCE

045366  016237  000000  002770  I.HDAL: MOV     RKCS1(R2),T.CS1    ;STORE CS1 TO CHECK CONTROLLER
                                                                   ;  ERROR
045374  032737  100000  002770          BIT     #CERR,T.CS1        ;CHECK IF CONTROLLER ERROR
045402  001422                          BEQ     5$                 ;NO, CHECK FOR ATTENTION

045404  005037  003044                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETE
045410  105037  003070                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
045414  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
045420  013765  002770  000016          MOV     T.CS1,P.CS1(R5)    ;STORE ERROR RKCS1
045426  004737  047430                  JSR     PC,I.CSTI          ;STORE CONTROLLER REGISTERS
045432  004037  046754                  JSR     R0,I.CCLR          ;CLEAR CONTROLLER
045436  046736                          I.RTRN                     ;ERROR RETURN
045440  004737  047740                  JSR     PC,R.ABNL          ;INDICATE ERROR RETURN
045444  000137  046736                  JMP     I.RTRN             ;RESTORE REGISTERS

045450  016537  000016  003004  5$:     MOV     RKASOF(R5),T.ASOF  ;STORE ATTENTION SUMMARY
045456  133737  003071  003005          BITB    INTMSK,T.ASOF+1    ;CHECK IF DRIVE ATTENTION IS SET
045464  001410                          BEQ     7$                 ;NO, CHECK IF READ ALL HEADERS
045466  005037  003044                  CLR     O.WAIT             ;CLEAR WAITING FOR COMMAND COMPLETION
045472  105037  003070                  CLRB    W.TIME             ;RESET TIMING ON DRIVE
045476  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT
045502  000137  045122                  JMP     I.ERRA             ;GO REPORT ERROR

045506  013701  003060          7$:     MOV     HDR.AD,R1          ;GET MAIN MEMORY ADDRESS
045512  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET FIRST WORD OF HEADER
045516  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET SECOND WORD OF HEADER
045522  016221  000024                  MOV     RKDB(R2),(R1)+     ;GET THIRD WORD OF HEADER
045526  010137  003060                  MOV     R1,HDR.AD          ;STORE ADDRESS FOR NEXT HEADER
045532  016237  000010  002772          MOV     RKCS2(R2),T.CS2    ;STORE CS2 TO CHECK FOR DATA LATE
045540  032737  100000  002772          BIT     #DLT,T.CS2         ;CHECK FOR DATA LATE
045546  001055                          BNE     35$                ;YES, REPORT ERROR
045550  005337  003062                  DEC     HDR.CT             ;DECREMENT NUMBER OF HEADER YET TO READ
045554  001026                          BNE     25$                ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
045556  005037  003044                  CLR     O.WAIT             ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
045562  005037  003104                  CLR     W.DRV              ;CLEAR TIME OUT COUNT FOR THIS DRIVE
045566  105037  003070                  CLRB    W.TIME             ;CLEAR WATCH DOG TIME ON THIS DRIVE
045572  012762  000003  000026          MOV     #3,RKMR1(R2)       ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
045600  112737  000001  002770          MOVB    #DR.SEL,T.CS1      ;LOAD SELECT COMMAND
045606  004037  047036                  JSR     R0,I.ISSU          ;GET SECTOR COUNT
045612  046736                          I.RTRN                     ;ERROR RETURN
045614  013765  003016  000056          MOV     T.MR3,P.B11(R5)    ;LOAD SECTOR COUNT
```

```
045622  004737  047752                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
045626  000137  046736                  JMP     I.RTRN          ;RESTORE REGISTERS

045632  016562  000002  000020  25$:    MOV     P.CYLN(R5),RKDCY(R2) ;LOAD CYLINDER ADDRESS REGISTER
045640  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
045646  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
045654  042765  165777  000016          BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
                                                ;  DRIVE TYPE
045662  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
045670  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
045676  000137  046736                  JMP     I.RTRN          ;RESTORE REGISTERS

045702  052737  000400  003042  35$:    BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
045710  004737  047764                  JSR     PC,R.CONT       ;REPORT ERROR
045714  000137  046736                  JMP     I.RTRN          ;RESTORE REGISTERS

                                .SBTTL  *DRIVE ATTENTION SCANNER

045720  016237  000000  002770  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
                                                ;  REGISTER 1 FOR COMPARISON
045726  032737  100000  002770          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
045734  001441                          BEQ     5$              ;NO, CHECK IF ATTENTION

                                        ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
045736  032737  164000  002772          BIT     #DLT!WCE!UPE!NEM,T.CS2

045744  001007                          BNE     1$              ;INDICATE ERROR
045746  016237  000014  003006          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER

                                ;       CHECK FOR DATA TRANSFER ERROR TYPE
045754  032737  125700  003006          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER

045762  001407                          BEQ     2$              ;NO DATA TRANSFER ERROR

045764  052737  000010  003042  1$:     BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
045772  004737  047764                  JSR     PC,R.CONT       ;REPORT ERROR
045776  000137  046736                  JMP     I.RTRN          ;RESTORE REGISTERS

046002  013704  002772          2$:     MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
046006  042704  177770                  BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
046012  105037  003070                  CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
046016  005037  003104                  CLR     W.DRV           ;RESET TIMER VALUE
046022  013705  003102                  MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5

                                ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
                                ;         IN PROGRAM DEVICE STATUS REGISTER
046026  042765  000006  000014          BIC     #DRVPOS!DRVPDT,P.PRST(R5)

046034  000137  045130                  JMP     I.ERRC          ;GO REPORT ERROR

046040  032737  040000  002770  5$:     BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
046046  001002                          BNE     6$              ;YES, PROCESS INTERRUPT
046050  000137  046736                  JMP     I.RTRN          ;RESTORE REGISTERS

046054  016237  000016  003004  6$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
046062  105737  003005                  TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
046066  001007                            BNE     7$                  ;YES GO PROCESS INTERRUPT
046070  052737  000002  003042            BIS     #E.NOAT,E.CONT      ;SET NO ATTENTION IN ATTENTION SUMMARY
046076  004737  047764                    JSR     PC,R.CONT           ;GO REPORT ERROR
046102  000137  046736                    JMP     I.RTRN              ;GO RESTORE REGISTERS

046106  133737  003071  003005  7$:       BITB    INTMSK,T.ASOF+1     ;CHECK IF DESIRED INTERRUPT
046114  001007                            BNE     8$                  ;YES, GO PROCESS IT
046116  052737  000004  003042            BIS     #E.UATT,E.CONT      ;SET UNSOLICATED ATTENTION
046124  004737  047764                    JSR     PC,R.CONT           ;GO REPORT ERROR
046130  000137  046736                    JMP     I.RTRN              ;GO RESTORE REGISTERS

046134  013705  003102          8$:       MOV     PBLKT,R5            ;STORE PARAMETER BLOCK TABLE
046140  116504  000000                    MOVB    P.DRVN(R5),R4       ;STORE DRIVE NUMBER
046144  032765  020000  000014            BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
046152  001402                            BEQ     11$                 ;NO, CONTINUE
046154  000137  046636                    JMP     I.UNLD              ;SERVICE DRIVE IN POSITION AFTER ERROR

046160  042765  000002  000014  11$:      BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
046166  005062  000026                    CLR     RKMR1(R2)           ;CLEAR MAINTENANCE REGISTER 1
046172  112737  000021  002770            MOVB    #DR.SEL,T.CS1       ;LOAD COMMAND
046200  004037  047036                    JSR     R0,I.ISSU           ;SELECT DRIVE WITH ATTENTION HIGH
046204  046736                            I.RTRN                      ;ERROR RETURN
046206  013765  003016  000042            MOV     T.MR3,P.B00(R5)     ;STORE STATUS BYTE 00 MESS B
046214  032765  000200  000042            BIT     #S.FLT,P.B00(R5)    ;CHECK IF DRIVE FAULT
046222  001401                            BEQ     12$                 ;NO, CHECK FOR DRIVE STATUS CHANGE
046224  000461                            BR      I.AERR              ;PROCESS ERROR

046226  013765  003014  000040  12$:      MOV     T.MR2,P.A00(R5)     ;STORE MAINTENANCE REGISTER 2
046234  032765  040000  000040            BIT     #S.DSC,P.A00(R5)    ;CHECK FOR DRIVE STATUS CHANGE
046242  001004                            BNE     13$                 ;YES, PROCESS DRIVE STATUS CHANGE
046244  052765  004000  000014            BIS     #NODSC,P.PRST(R5)   ;SET NO DRIVE STATUS CHANGE
046252  000446                            BR      I.AERR              ;PROCESS ERROR

046254  112737  000005  002770  13$:      MOVB    #DR.CLR,T.CS1       ;LOAD COMMAND
046262  004037  047036                    JSR     R0,I.ISSU           ;CLEAR DRIVE STATUS CHANGE
046266  046736                            I.RTRN                      ;ERROR RETUN
046270  013765  003004  000032            MOV     T.ASOF,P.ASOF(R5)   ;STORE ATTENTION SUMMARY
046276  133765  003071  000033            BITB    INTMSK,P.ASOF+1(R5) ;CHECK IF ATTENTION RESET
046304  001407                            BEQ     15$                 ;YES, CONTINUE INTERRUPT PROCESSING
046306  052737  000020  003042            BIS     #E.CLAT,E.CONT      ;SET ATTENTION DID NOT RESET
                                                                      ;  WITH DRIVE CLEAR
046314  004737  047764                    JSR     PC,R.CONT           ;FLAG ERROR
046320  000137  046736                    JMP     I.RTRN              ;RESTORE REGISTERS

046324  013765  003014  000040  15$:      MOV     T.MR2,P.A00(R5)     ;STORE MAINTENANCE REGISTER 2
046332  032765  040000  000040            BIT     #S.DSC,P.A00(R5)    ;CHECK IF DRIVE STATUS CHANGE
                                                                      ;  RESET
046340  001404                            BEQ     16$                 ;YES, CONTINUE INTERRUPT PROCESSING
046342  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5)  ;SET DRIVE STATUS CHANGE DID NOT CLEAR
046350  000407                            BR      I.AERR              ;GO PROCESS ERROR

046352  105037  003070          16$:      CLRB    W.TIME              ;RESET TIMING ON THIS DRIVE
046356  005037  003104                    CLR     W.DRV               ;CLEAR DRIVE TIMING COUNT
046362  004737  047752                    JSR     PC,R.NORM           ;REPORT SUCCESSFUL COMMAND COMPLETETION
046366  000563                            BR      I.RTRN              ;RESTORE REGISTERS
```

```
                                        .SBTTL   *ATTENTION ERROR HANDLER
                                
046370  042765  000004  000014  I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
                                                                   ;  OF DATA TRANSFER
046376  105037  003070                  CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
046402  005037  003104                  CLR     W.DRV              ;RESET WATCH-DOG TIME
046406  042765  177741  000016          BIC     #177741,P.CS1(R5)  ;KEEP COMMAND ISSUED
046414  042737  000036  002770          BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
046422  053765  002770  000016          BIS     T.CS1,P.CS1(R5)    ;MAKE GOOD MESSAGE
046430  013765  002772  000020          MOV     T.CS2,P.CS2(R5)    ;STORE CONTROLLER REGISTERS
046436  013765  002774  000022          MOV     T.WCR,P.WCR(R5)
046444  013765  002776  000024          MOV     T.BA,P.BAR(R5)
046452  013765  003000  000026          MOV     T.DA,P.DTS(R5)
046460  013765  003002  000030          MOV     T.DC,P.DCYL(R5)
046466  013765  003004  000032          MOV     T.ASOF,P.ASOF(R5)
046474  013765  003006  000034          MOV     T.ER,P.ER(R5)
046502  013765  003010  000036          MOV     T.DS,P.DS(R5)
046510  004037  047512                  JSR     R0,I.STAT          ;GATHER DRIVE STATUS
046514  046736                          I.RTRN                     ;ERROR RETURN
046516  112737  000005  002770          MOVB    #DR.CLR,T.CSI      ;LOAD COMMAND
046524  004037  047036                  JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
046530  046736                          I.RTRN                     ;ERROR RETURN
046532  133737  003071  003005          BITB    INTMSK,T.ASOF+1    ;CHECK IF ATTENTION RESET
046540  001407                          BEQ     2$                 ;YES, FLAG DRIVE ERROR
046542  052737  000020  003042          BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
046550  004737  047764                  JSR     PC,R.CONT          ;REPORT ERROR
046554  000137  046736                  JMP     I.RTRN             ;RESTORE REGISTERS

046560  032765  000020  000014  2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
046566  001017                          BNE     10$                ;YES, REPORT ERROR
046570  032737  020000  003014          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
046576  001413                          BEQ     10$                ;NO, REPORT ERROR
046600  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
046606  113737  003071  003070          MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
046614  013737  003054  003104          MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
046622  000137  046736                  JMP     I.RTRN             ;RESTORE REGISTERS

046626  004737  047740          10$:    JSR     PC,R.ABNL          ;REPORT ERROR
046632  000137  046736                  JMP     I.RTRN             ;RESTORE REGISTERS

                                        .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD

046636  052765  020000  000014  I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
046644  112737  000005  002770          MOVB    #DR.CLR,T.CSI      ;LOAD IN DRIVE CLEAR
046652  004037  047036                  JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
046656  046736                          I.RTRN                     ;ERROR RETURN
046660  136437  003071  003005          BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
046666  001406                          BEQ     15$                ;YES, CONTINUE
046670  012737  000020  003042          MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
046676  004737  047764                  JSR     PC,R.CONT          ;REPORT ERROR
046702  000415                          BR      I.RTRN             ;RESTORE REGISTERS

046704  032737  040000  003014  15$:    BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
046712  001403                          BEQ     20$                ;YES, CONTINUE
046714  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
046722  105037  003070          20$:    CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

```
:0046  046726  005037  003104                CLR     W.DRV           ;CLEAR TIME COUNT
:0047  046732  004737  004440                JSR     PC,P.ABNL       ;REPORT ERROR
:0048
:0049  046736  012600                I.RTRN: MOV     (SP)+,R0        ;RESTORE R0
:0050  046742  012601                        MOV     (SP)+,R1        ;RESTORE R1
:0051  046746  012602                        MOV     (SP)+,R2        ;RESTORE R2
:0052  046752  012603                        MOV     (SP)+,R3        ;RESTORE R3
:0053  046756  012604                        MOV     (SP)+,R4        ;RESTORE R4
:0054  046762  012605                        MOV     (SP)+,R5        ;RESTORE R5
:0055  046766  000002                        RTI                     ;RETURN
:0056
```

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

046754  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
046762  016237  000000  002770          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
046770  032737  100000  002770          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
046776  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
047000  052737  000001  003042          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
047006  004737  047764                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
047012  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
047014  000200                          RTS     R0              ;RETURN

047016  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
047024  112737  177777  003064          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
047032  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
047034  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
                                ;*
                                ;;*****************************************************************

047036  013746  002770        I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
047042  005037  002772                CLR     T.CS2           ;CLEAR TEMPORARY CS2
047046  116537  000000  002772        MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
047054  013762  002772  000010        MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
047062  116537  000007  002771        MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
047070  142737  177753  002771        BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
                                                                ;  FORMAT AND DRIVE TYPE
047076  013762  002770  000000        MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
047104  105762  000000        1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
047110  100375                        BPL     1$
047112  004737  047260                JSR     PC,I.STOR       ;GO STORE REGISTERS
047116  032737  100000  002770        BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
047124  001437                        BEQ     5$              ;NO, RETURN
047126  032737  001000  002772        BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
047134  001406                        BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
047136  052737  100000  003042        BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
047144  004737  047764                JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
047150  000440                        BR      10$             ;RETURN

                                ;CHECK IF ANY CONTROLLER ERROR IS SET
047152  032737  024000  002770  2$:   BIT     #CTO!SPAR,T.CS1
047160  001027                        BNE     7$
047162  032737  176400  002772        BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
047170  001023                        BNE     7$
047172  032737  131761  003006        BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
047200  001017                        BNE     7$

047202  122716  000005                CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
10148  047206  001003                        BNE     3$                ;NO, DO NOT SET DRIVE HARD ERROR
10149  047210  052765  000020  000014        BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
10150  047216  004037  046754          3$:   JSR     R0,I.CCLR         ;GO ISSUE A CONTROLLER CLEAR
10151  047222  047252                        10$                       ;ERROR RETURN
10152  047224  012762  000100  000000  5$:   MOV     #IE,RKCS1(R2)     ;SET INTERRUPT ENABLE
10153  047232  005726                        TST     (SP)+             ;ADJUST STACK
10154  047234  005720                        TST     (R0)+             ;ADJUST R0 FOR NORMAL RETURN
10155  047236  000200                        RTS     R0                ;RETURN
10156
10157  047240  052737  001000  003042  7$:   BIS     #E.CERR,E.CONT    ;SET CONTROLLER ERROR DURING
10158                                                                  ;  DRIVER SERVICING
10159  047246  004737  047764                JSR     PC,R.CONT         ;REPORT ERROR
10160  047252  005726                  10$:  TST     (SP)+             ;ADJUST STACK
10161  047254  011000                        MOV     (R0),R0           ;ADJUST R0 FOR ERROR RETURN
10162  047256  000200                        RTS     R0                ;RETURN
```

```
                        .SBTTL   *STORE RK611 UNIBUS REGISTERS

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

010180  047260  016237  000000  002770  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
010181  047266  016237  000010  002772          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
010182  047274  016237  000002  002774          MOV     RKWC(R2),T.WCR
010183  047302  016237  000004  002776          MOV     RKBA(R2),T.BA
010184  047310  016237  000006  003000          MOV     RKDA(R2),T.DA
010185  047316  016237  000012  003010          MOV     RKDS(R2),T.DS
010186  047324  016237  000014  003006          MOV     RKER(R2),T.ER
010187  047332  016237  000016  003004          MOV     RKASOF(R2),T.ASOF
010188  047340  016237  000020  003002          MOV     RKDCYL(R2),T.DC
010189  047346  016237  000026  003012          MOV     RKMR1(R2),T.MR1
010190  047354  016237  000034  003014          MOV     RKMR2(R2),T.MR2
010191  047362  016237  000036  003016          MOV     RKMR3(R2),T.MR3
010192  047370  016237  000030  003020          MOV     RKECPS(R2),T.POS
010193  047376  016237  000032  003022          MOV     RKECPT(R2),T.PAT
010194  047404  000207                          RTS     PC              ;RETURN
```

```
10195                                   .SBTTL   *STORE CONTROLLER STATUS
10196
10197                                   ;;*****************************************************************
10198                                   ;*
10199                                   ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
10200                                   ;*      THE FOLLOWING REGISTERS WILL BE STORED:
10201                                   ;*
10202                                   ;*              COMMAND AND STATUS REGISTER 2
10203                                   ;*              WORD COUNT REGISTER
10204                                   ;*              BUS ADDRESS REGISTER
10205                                   ;*              DESIRED TRACK AND SECTOR
10206                                   ;*              STATUS REGISTER
10207                                   ;*              ERROR REGISTER
10208                                   ;*              ATTENTION SUMMARY/OFFSET REGISTER
10209                                   ;*              CYLINDER ADDRESS REGISTER
10210                                   ;*
10211                                   ;*CALL  JSR     PC,I.CSTS
10212                                   ;*      RETURN
10213                                   ;*
10214                                   ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
10215                                   ;*
10216                                   ;*
10217                                   ;*              REGISTER                CONTENTS
10218                                   ;*              --------                --------
10219                                   ;*
10220                                   ;*              R2                      RK06 BASE ADDRESS
10221                                   ;*              R5                      ADDRESS OF PARAMETER BLOCK
10222                                   ;*
10223                                   ;;*****************************************************************
10224
10225  047406  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
10226                                                                          ;OF LAST COMMAND ISSUED
10227  047414  042737  000036  002770          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
10228  047422  053765  002770  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
10229  047430  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
10230  047436  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
10231  047444  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
10232  047452  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
10233  047460  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
10234  047466  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
10235  047474  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
10236                                                                          ;  OFFSET
10237  047502  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
10238  047510  000207                          RTS     PC                      ;RETURN
10239
```

```
10240                                   .SBTTL   *GATHER DRIVE STATUS
10241
10242                                   ;;*****************************************************************
10243                                   ;*
10244                                   ;*      THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
10245                                   ;*      BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
10246                                   ;*      HAS PREVIOUSLY BEEN SEIZED.  IT RUNS AT PRIORITY 7.
10247                                   ;*
10248                                   ;*CALL  JSR     R0,I.STAT
10249                                   ;*      <ADDRESS OF ERROR RETURN>
10250                                   ;*      RETURN
10251                                   ;*
10252                                   ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
10253                                   ;*
10254                                   ;*
10255                                   ;*              REGISTER                CONTENTS
10256                                   ;*              --------                --------
10257                                   ;*
10258                                   ;*              R2                      RK06 BASE ADDRESS
10259                                   ;*              R5                      ADDRESS OF PARAMETER BLOCK
10260                                   ;*
10261                                   ;*      ROUTINES USED:
10262                                   ;*              I.ISSU
10263                                   ;*
10264                                   ;;*****************************************************************
10265
10266  047512  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
10267                                                                          ;  FOR STATUS BYTE 01
10268  047520  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10269  047526  004037  047036                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
10270  047532  047722                          3$                              ;ERROR RETURN
10271  047534  013765  003014  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
10272  047542  013765  003016  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
10273  047550  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
10274                                                                          ;  FOR STATUS BYTE 10
10275  047556  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10276  047564  004037  047036                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
10277  047570  047722                          3$                              ;ERROR RETURN
10278  047572  013765  003014  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
10279  047600  013765  003016  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
10280  047606  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
10281                                                                          ;  FOR STATUS BYTE 11
10282  047614  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10283  047622  004037  047036                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
10284  047626  047722                          3$                              ;ERROR RETURN
10285  047630  013765  003014  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
10286  047636  013765  003016  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
10287  047644  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
10288                                                                          ;  FOR STATUS BYTE 00
10289  047650  112737  000001  002770          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10290  047656  004037  047036                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
10291  047662  047722                          3$                              ;ERROR RETURN
10292  047664  013765  003014  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
10293  047672  013765  003016  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
10294  047700  032737  001000  003016          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
10295  047706  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
10296  047710  052737  002000  003042          BIS     #E.DPAR,E.CONT          ;INDICATE BAD PARITY DETECTED BY DRIVE
10297  047716  004737  047764                  JSR     PC,R.CONT               ;REPORT ERROR
10298  047722  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
10299  047724  000200                          RTS     R0                      ;RETURN
10300
10301  047726  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
10302  047734  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
10303  047736  000200                          RTS     R0                      ;RETURN
10304
```

B16


```
                                        .SBTTL  *COMMON DRIVER RETURNS

047740  105037  003071          R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
047744  004777  133066                  JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
047750  000207                          RTS     PC              ;RETURN

047752  105037  003071          R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
047756  004777  133052                  JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
047762  000207                          RTS     PC              ;RETURN

047764  105037  003071          R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
047770  105037  003070                  CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
047774  005037  003104                  CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
050000  004777  133034                  JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
050004  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
                        ;*
                        ;;*****************************************************************

 010350
 010351  050006  010546                C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
 010352  050010  010446                        MOV     R4,-(SP)        ;STORE R4 ON STACK
 010353  050012  010346                        MOV     R3,-(SP)        ;STORE R3 ON STACK
 010354  050014  010246                        MOV     R2,-(SP         ;STORE R2 ON STACK
 010355  050016  010146                        MOV     R1,-(SP)        ;STORE R1 ON STACK
 010356  050020  010046                        MOV     R0,-(SP)        ;STORE R0 ON STACK
 010357  050022  013746  177776                MOV     PS,-(SP)        ;STORE PSW ON STACK
 010358  050026  013737  003032  177776        MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
 010359  050034  017605  000016                MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
 010360  050040  062766  000002  000016        ADD     #2,16(SP)       ;ADJUST RETURN
 010361  050046  016504  000000                MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
 010362  050052  042704  177770                BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
 010363  050056  010537  003102                MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
 010364  050062  116437  003072  003071        MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
 010365  050070  116437  003072  003070        MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
 010366  050076  013737  003052  003104        MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME
 010367
 010368  050104  013702  003026                MOV     RKBAS,R2        ;LOAD R2 WITH RK06 ADDRESS BASE
 010369
 010370                                ;       RESET ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
 010371                                ;               DRIVE IN USE
 010372                                ;               WRITE FOR WRITE CHECK
 010373                                ;               NO CHECK
 010374                                ;               DROP DRIVE FROM TEST SEQUENCE
 010375                                ;               INHIBIT BUS ADDRESS INCREMENT
```

```
050110  042765  075176  000014          BIC     #^C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PR  (R5)

050116  010500                          MOV     R5,R0               ;STORE PARAMETER BLOCK ADDRESS
050120  062700  000016                  ADD     #P.CS1,R0           ;CALCULATE FIRST LOCATION TO BE CLEARED
050124  010501                          MOV     R5,R1               ;STORE PARAMETER BLOCK ADDRESS
050126  062701  000062                  ADD     #P.EPAT,R1          ;CALCULATE LAST LOCATION TO BE CLEARED

050132  005020                  1$:     CLR     (R0)+               ;CLEAR RETURN PARAMETER
050134  020001                          CMP     R0,R1               ;CHECK IF FINISHED
050136  101775                          BLOS    1$                  ;NO, CLEAR NEXT RETURN PARAMETER
050140  105037  003064                  CLRB    I.ISRL              ;CLEAR RELEASE OR INTERRUPT ISSUED
050144  010465  000020                  MOV     R4,P.CS2(R5)        ;STORE DRIVE NUMBER
050150  005062  000026                  CLR     RKMR1(R2)           ;CLEAR RK06 MAINTENANCE REGISTER 1
050154  132765  000040  000001          BITB    #BITS,P.CMND(R5)    ;CHECK IF SPECIAL COMMAND
050162  001402                          BEQ     3$                  ;NO, PROCESS
050164  000137  050700                  JMP     C.SPEC              ;JUMP TO SPECIAL COMMAND PROCESSOR

050170  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5)  ;CHECK IF POSITIONING COMMAND
                                                                    ;  START SPINDLE
                                                                    ;  RECALIBRATE
                                                                    ;  OFFSET
                                                                    ;  SEEK
                                                                    ;  UNLOAD

050176  101174                          BHI     25$                 ;NO, DRIVE COMMAND
                                                                    ;  SELECT DRIVE
                                                                    ;  PACK ACKNOWLEDGE
                                                                    ;  CLEAR

050200  122765  000117  000001          CMPB    #SEEK,P.CMND(R5)    ;CHECK IF DATA TRANSFER
050206  103540                          BLO     20$                 ;YES, DATA TRANSFER COMMAND
                                                                    ;  READ DATA
                                                                    ;  WRITE DATA
                                                                    ;  READ HEADER
                                                                    ;  WRITE HEADER
                                                                    ;  WRITE CHECK
050210  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
050216  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5)  ;SET DRIVE POSITIONING
050224  005037  003044                  CLR     O.WAIT              ;CLEAR WAIT FOR COMMAND
050230  122765  000117  000001          CMPB    #SEEK,P.CMND(R5)    ;CHECK IF SEEK
050236  001007                          BNE     5$                  ;NO, CHECK FOR OFFSET
050240  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
050246  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
050254  000431                          BR      8$                  ;GO ISSUE COMMAND

050256  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5)  ;CHECK IF OFFSET
050264  001007                          BNE     6$                  ;NO, CHECK FOR UNLOAD
050266  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
050274  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
050302  000416                          BR      8$                  ;GO ISSUE COMMAND

050304  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5)  ;CHECK IF START SPINDLE
050312  001003                          BNE     7$                  ;NO, CHECK IF RECAL
050314  013737  003056  003104          MOV     W.MIN,W.DRV         ;LOAD WATCH DOG TIME FOR 1 MINUTE
050322  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5)   ;CHECK IF RECAL
050330  001003                          BNE     8$                  ;NO, CONTINUE
```

```
050332  013737  003054  003104          MOV     W.BSEC,W.DRV    ;LOAD RECAL TIME FOR 8 SECONDS
050340  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
050346  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
                                                ;  AND DRIVE TYPE
050354  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
050362  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
050370  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
050376  001533                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
050400  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
050406  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
050414  004737  044356          10$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
050420  016237  000000  002770          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
050426  032737  000200  002770          BIT     #RDY,T.CS1      ;WAIT FOR READY
050434  001767                          BEQ     10$
050436  032737  100000  002770          BIT     #CERR,T.CS1     ;CHECK FOR ERROR
050444  001011                          BNE     15$             ;YES, GIVE NORMAL RETURN
050446  004737  044356          11$:    JSR     PC,W.WTCH       ;CALL WATCH DOG TIMER
050452  016237  000016  003004          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
050460  133737  003071  003005          BITB    INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
050466  001767                          BEQ     11$             ;WAIT FOR DRIVE INTERRUPT
050470  105037  003070          15$:    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
050474  005037  003104                  CLR     W.DRV           ;CLEAR DRIVE TIMING COUNT
050500  004737  047752                  JSR     PC,R.NORM       ;INDICATE COMMAND IS FINISHED
050504  000137  051660                  JMP     C.RTRN          ;RESTORE REGISTERS

050510  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
050516  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
050524  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
050532  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
050540  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
050546  001010                          BNE     25$             ;NO, GO ISSUE THE COMMAND
050550  032765  000200  000014          BIT     #W.WCK,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
050556  001404                          BEQ     25$             ;NO, GO ISSUE THE COMMAND
050560  012765  000123  000016          MOV     #WRDATA,P.CS1(R5) ;ISSUE WRITE COMMAND
050566  000406                          BR      26$             ;GO ISSUE COMMAND

050570  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
050576  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
050604  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
050612  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
                                                ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
                                                ;  BITS 16-17
050620  010537  003044                  MOV     R5,O.WAIT       ;LOAD WAITING FOR COMMAND
050624  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
050632  001403                          BEQ     27$             ;NO, LOAD CS2
050634  052765  000020  000020          BIS     #BAI,P.CS2(R5)  ;SET INHIBIT BUS ADDRESS INCREMENT
050642  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
050650  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
050656  001403                          BEQ     30$             ;NO, SKIP CLEAR OF INTERRUPT ENABLE
050660  042765  000100  000016          BIC     #IE,P.CS1(R5)   ;CLEAR INTERRUPT ENABLE
050666  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
050674  000137  051660                  JMP     C.RTRN          ;RESTORE REGISTERS


                                .SBTTL  *SPECIAL COMMAND PROCESSING

050700  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
 10488  050706  001132                        BNE     10$             ;NO, PROCESS OTHER COMMANDS
 10489  050710  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
 10490  050716  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 10491  050724  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
 10492                                                                  ;  AND DRIVE TYPE
 10493  050732  112765  000001  000016        MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
 10494  050740  016562  000016  000000        MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 10495  050746  004737  044356          2$:   JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
 10496  050752  016265  000000  000016        MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
 10497  050760  032765  000200  000016        BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
 10498  050766  001767                        BEQ     2$
 10499  050770  004737  047430                JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
 10500  050774  016265  000034  000040        MOV     RKMR2(R2),P.ADD(R5) ;STORE STATUS BYTE OO MESSAGE A
 10501  051002  016265  000036  000042        MOV     RKMR3(R2),P.BDD(R5) ;STORE STATUS BYTE OO MESSAGE B
 10502  051010  032765  100000  000016        BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
 10503  051016  001436                        BEQ     6$              ;NO, GATHER DRIVE STATUS
 10504  051020  105037  003070                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
 10505  051024  005037  003104                CLR     W.DRV           ;CLEAR WATCH DOG COUNT
 10506  051030  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10507  051036  001043                        BNE     8$              ;YES, INDICATE NORMAL RETURN
 10508  051040  032765  001000  000020        BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
 10509  051046  001043                        BNE     9$              ;YES, INDICATE CONTROLLER ERROR
 10510  051050  004037  046754                JSR     R0,I.CCLR       ;CLEAR ERROR
 10511  051054  051660                        C.RTRN                  ;ERROR RETURN
 10512  051056  032765  010400  000020        BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
 10513  051064  001007                        BNE     5$              ;REPORT ERROR
 10514  051066  032765  000001  000036        BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
 10515  051074  001003                        BNE     5$              ;YES, REPORT ERROR
 10516  051076  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
 10517  051104  004737  047740          5$:   JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
 10518  051110  000137  051660                JMP     C.RTRN          ;RESTORE REGISTERS
 10519
 10520  051114  004037  047512          6$:   JSR     R0,I.STAT       ;GATHER DRIVE STATUS
 10521  051120  051660                        C.RTRN                  ;ERROR RETURN
 10522  051122  105037  003070                CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
 10523  051126  005037  003104                CLR     W.DRV           ;RESET WATCH-DOG TIME
 10524  051132  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10525  051140  001402                        BEQ     8$              ;NO, REPORT ERROR
 10526  051142  005062  000000                CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
 10527  051146  004737  047752          8$:   JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
 10528  051152  000137  051660                JMP     C.RTRN          ;RESTORE REGISTERS
 10529
 10530  051156  052737  100000  003042  9$:   BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
 10531  051164  004737  047764                JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
 10532  051170  000137  051660                JMP     C.RTRN
 10533
 10534  051174  122765  000140  000001  10$:   CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
 10535  051202  001040                        BNE     13$             ;NO, CHECK IF READ ALL HEADERS
 10536  051204  010537  003044                MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
 10537                                                                  ;  WAIT FOR COMMAND
 10538  051210  052765  000010  000020        BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
 10539  051216  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
 10540  051224  112737  000001  003064        MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
 10541  051232  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
 10542  051240  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
 10543                                                                  ;  AND DRIVE TYPE
```

```
 10544  051246  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
 10545  051254  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10546  051262  001403                          BEQ     11$               ;NO, DO NOT RESET INTERRUPT ENABLE
 10547  051264  042765  000100  000016          BIC     #IE,P.CS1(R5)     ;RESET INTERRUPT ENABLE
 10548  051272  016562  000016  000000  11$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 10549  051300  000137  051660                  JMP     C.RTRN            ;RESTORE REGISTERS
 10550
 10551  051304  122765  000164  000001  13$:    CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 10552  051312  001053                          BNE     30$               ;NO, CHECK IF CONTROLLER CLEAR
 10553  051314  010537  003044                  MOV     R5,@.WAIT         ;SET WAITING FOR COMMAND COMMPLETION
 10554  051320  016537  000010  003060          MOV     P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
 10555  051326  132765  000020  000007          BITB    #8.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
 10556  051334  001404                          BEQ     14$               ;YES, LOAD 22 IN HEADER COUNT
 10557  051336  012737  000024  003062          MOV     #20.,HDR.CT       ;LOAD 20 IN SECTOR COUNT
 10558  051344  000403                          BR      22$               ;GO ISSUE READ HEADER COMMAND
 10559
 10560  051346  012737  000026  003062  14$:    MOV     #22.,HDR.CT       ;LOAD 22 IN SECTOR COUNT
 10561  051354  016562  000002  000020  22$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
 10562  051362  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
 10563  051370  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
 10564  051376  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 10565  051404  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
 10566                                                  ;  AND FORMAT
 10567  051412  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
 10568  051420  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10569  051426  001027                          BNE     34$               ;YES, INDICATE ILLEGAL DRIVER COMMAND
 10570  051430  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
 10571  051436  000137  051660                  JMP     C.RTRN            ;RESTORE REGISTERS
 10572
 10573  051442  122765  000176  000001  30$:    CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER CLEAR
 10574  051450  001012                          BNE     32$               ;NO, CHECK IF SUBSYSTEM CLEAR
 10575  051452  004737  046754                  JSR     R0,I.CCLR         ;CLEAR CONTROLLER
 10576  051456  000163                          C.RTRN                    ;ERROR RETURN
 10577  051460  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10578  051466  001472                          BEQ     40$               ;NO, INDICATE NORMAL RETURN
 10579  051470  005062  000000                  CLR     RKCS1(R2)         ;RESET INTERRUPT ENABLE
 10580  051474  000467                          BR      40$               ;INDICATE NORMAL RETURN
 10581
 10582  051476  122765  000177  000001  32$:    CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
 10583  051504  001406                          BEQ     36$               ;YES, CLEAR SUBSYSTEM
 10584  051506  052737  000100  003042  34$:    BIS     #E.ILLD,E.CONT    ;SET ILLEGAL DRIVER COMMAND
 10585  051514  004737  047764                  JSR     PC,R.CONT         ;REPORT ERROR
 10586  051520  000457                          BR      C.RTRN            ;RESTORE REGISTERS
 10587
 10588  051522  012762  000040  000010  36$:    MOV     #SCLR,RKCS2(R2)   ;ISSUE SUBSYSTEM CLEAR
 10589  051530  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
 10590  051536  032765  100000  000016          BIT     #CERR,P.CS1(R5)   ;CLEAR IF CONTROLLER ERROR RESET
 10591  051544  001406                          BEQ     37$               ;NO, FINISH COMMAND
 10592  051546  052737  000001  003042          BIS     #BIT0,E.CONT      ;SET CLEAR SUBSYSTEM DID NOT CLEAR
 10593                                                  ;  CONTROLLER ERROR
 10594  051554  004737  047764                  JSR     PC,R.CONT         ;REPORT ERROR
 10595  051560  000437                          BR      C.RTRN            ;RESTORE REGISTERS
 10596
 10597  051562  013746  003050          37$:    MOV     W.MILI,-(SP)      ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
 10598                                                  ;  TO DISAPPEAR
 10599  051566  016265  000000  000016  38$:    MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
10630  051574  032765  040000  000016          BIT     #DI.P.CSI(R5)   ;CHECK IF ATTENTIONS CLEARED
10631  051602  001411                          BEQ     39$             ;YES, FINISH COMMAND
10632  051604  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
10633  051606  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
10634  051610  005726                          TST     (SP)+           ;ADJUST STACK
10635  051612  052737  000040  003042          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10636                                                                  ;  DRIVE ATTENTIONS
10637  051620  004737  047764                  JSR     PC,R.CONT       ;REPORT ERROR
10638  051624  000415                          BR      C.RTRN          ;RESTORE REGISTER
10639
10640  051626  005726                  39$:    TST     (SP)+           ;ADJUST STACK
10641  051630  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10642  051636  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
10643  051640  112737  177777  003064          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
10644  051646  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10645  051654  004737  047752                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
10646
10647  051660  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
10650  051664  012600                          MOV     (SP)+,R0        ;RESTORE R0
10651  051666  012601                          MOV     (SP)+,R1        ;RESTORE R1
10652  051670  012602                          MOV     (SP)+,R2        ;RESTORE R2
10653  051672  012603                          MOV     (SP)+,R3        ;RESTORE R3
10654  051674  012604                          MOV     (SP)+,R4        ;RESTORE R4
10655  051676  012605                          MOV     (SP)+,R5        ;RESTORE R5
10656  051700  000207                          RTS     PC              ;RETURN
10657                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
10660
10661                                  ;;*****************************************************************
10662                                  ;*
10663                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10664                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
10665                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10666                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
10667                                  ;*
10670                                  ;*CALL
10671                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
10672                                  ;*      JSR     PC,OCTBIN
10673                                  ;*      <ADDRESS OF ERROR RETURN>
10674                                  ;*      RETURN
10675                                  ;*
10676                                  ;;*****************************************************************
10677
10700  051702  010046          OCTBIN: MOV     R0,-(SP)        ;SAVE R0
10701  051704  010146                  MOV     R1,-(SP)        ;SAVE R1
10702  051706  010246                  MOV     R2,-(SP)        ;SAVE R2
10703  051710  016600  000010          MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
10704  051714  005001                  CLR     R1              ;CLEAR DATA WORDS
10705  051716  005002                  CLR     R2
10706  051720  112046          2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
10707  051722  001423                  BEQ     3$              ;IF ZERO GET OUT
10710  051724  121627  000054          CMPB    (SP),#',        ;CHECK IF COMMA
10711  051730  001420                  BEQ     3$              ;IF COMMA GET OUT
10712  051732  122716  000060          CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
10713  051736  003030                  BGT     4$              ;  AN OCTAL DIGIT
10714  051740  122716  000067          CMPB    #'7,(SP)
10715  051744  002425                  BLT     4$
```

```
10656  051746  006301                        ASL     R1              ; *2
10657  051750  006102                        ROL     R2
10658  051752  006301                        ASL     R1              ; *4
10659  051754  006102                        ROL     R2
10660  051756  006301                        ASL     R1              ; *8
10661  051760  006102                        ROL     R2
10662  051762  042716  177770                BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
10663  051766  062601                        ADD     (SP)+,R1        ;ADD THIS DIGIT
10664  051770  000753                        BR      2$              ;LOOP
10665  051772  005726                3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
10666  051774  010166  000010                MOV     R1,10(SP)       ;SAVE RESULT
10667  052000  010237  052034                MOV     R2,$HIOCT
10668  052004  012602                        MOV     (SP)+,R2        ;RESTORE R2
10669  052006  012601                        MOV     (SP)+,R1        ;RESTORE R1
10670  052010  012600                        MOV     (SP)+,R0        ;RESTORE R0
10671  052012  062716  000002                ADD     #2,(SP)         ;ADJUST RETURN
10672  052016  000207                        RTS     PC              ;RETURN
10673
10674  052020  005726                4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
10675  052022  012602                        MOV     (SP)+,R2        ;RESTORE R2
10676  052024  012601                        MOV     (SP)+,R1        ;RESTORE R1
10677  052026  012600                        MOV     (SP)+,R0        ;RESTORE R0
10678  052030  013616                        MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
10679  052032  000207                        RTS     PC              ;GO PROCESS ERROR
10680  052034  000000                $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
10681                                 .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
10682
10683                                 ;;*****************************************************************
10684                                 ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
10685                                 ;*WITH A RANGE OF 0 TO 2(+33)-1.
10686                                 ;*CALL:
10687                                 ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
10688                                 ;*      RETURN                  ;;RETURN HERE THE RANDOM
10689                                 ;*                              ;;NUMBER WILL BE IN
10690                                 ;*                              ;;$HINUM,$LONUM
10691
10692  052036                        $RAND:
10693  052036  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10694  052040  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10695  052042  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10696  052044  013700  052136                MOV     $LONUM,R0       ;;SET R0 WITH LOW
10697  052050  013701  052134                MOV     $HINUM,R1       ;;SET R1 WITH HIGH
10698  052054  012702  177771                MOV     #-7,R2          ;;SET SHIFT COUNT
10699  052060  006300                1$:     ASL     R0              ;;SHIFT R0 LEFT AND
10700  052062  006101                        ROL     R1              ;;ROTATE CARRY INTO R1 AND
10701  052064  005202                        INC     R2              ;;CHECK FOR DONE
10702  052066  001374                        BNE     1$              ;;CONTINUE SHIFT LOOP
10703  052070  063700  052136                ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
10704  052074  005501                        ADC     R1              ;;PROPOGATE CARRY
10705  052076  063701  052134                ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
10706  052102  062700  001057                ADD     #1057,R0        ;;ADD LOW CONSTANT
10707  052106  005501                        ADC     R1              ;;PROPOGATE CARRY
10708  052110  062701  047401                ADD     #47401,R1       ;;ADD HIGH CONSTANT
10709  052114  010037  052136                MOV     R0,$LONUM       ;;SAVE R0
10710  052120  010137  052134                MOV     R1,$HINUM       ;;SAVE R1
10711  052124  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
10712  052126  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
10713  052130  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
10714  052132  000207                      RTS     PC              ;;RETURN
10715  052134  176543              $HINUM: .WORD   176543
10716  052136  123456              $LONUM: .WORD   123456
10717                               .SBTTL  TYPE ROUTINE
10718
10719                               ;;*****************************************************************
10720                               ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10721                               ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10722                               ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10723                               ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10724                               ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10725                               ;*
10726                               ;*CALL:
10727                               ;*1) USING A TRAP INSTRUCTION
10728                               ;*      TYPE    .MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
10729                               ;*OR
10730                               ;*      TYPE
10731                               ;*      MESADR
10732                               ;*
10733
10734  052140  105737  001157      $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
10735  052144  100002                      BPL     1$              ;;BR IF YES
10736  052146  000000                      HALT                    ;;HALT HERE IF NO TERMINAL
10737  052150  000430                      BR      3$              ;;LEAVE
10738  052152  010046              1$:     MOV     R0,-(SP)        ;;SAVE R0
10739  052154  017600  000002              MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
10740  052160  122737  000001  001340      CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
10741  052166  001011                      BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
10742  052170  132737  000100  001341      BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
10743  052176  001405                      BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
10744  052200  010037  052210              MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
10745  052204  004737  054636              JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
10746  052210  000000              61$:    .WORD   0               ;;MESSAGE ADDRESS
10747  052212  132737  000040  001341  62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
10748  052220  001003                      BNE     60$             ;;YES,SKIP TYPE OUT
10749  052222  112046              2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
10750  052224  001005                      BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
10751  052226  005726                      TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
10752  052230  012600              60$:    MOV     (SP)+,R0        ;;RESTORE R0
10753  052232  062716  000002      3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
10754  052236  000002                      RTI                     ;;RETURN
10755  052240  122716  000011      4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
10756  052244  001430                      BEQ     8$
10757  052246  122716  000200              CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
10758  052252  001006                      BNE     5$
10759  052254  005726                      TST     (SP)+           ;;POP  <CR><LF> EQUIV
10760  052256  104401                      TYPE                    ;;TYPE A CR AND LF
10761  052260  001315                      $CRLF
10762  052262  105037  052416              CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
10763  052266  000755                      BR      2$              ;;GET NEXT CHARACTER
10764  052270  004737  052352      5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
10765  052274  123726  001156      6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
10766  052300  001350                      BNE     2$              ;;IF NO GO GET NEXT CHAR.
10767  052302  013746  001154              MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
10768                                                   ;;AND THE NULL CHAR.
10769  052306  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
10770  052312  002770                  BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
10771  052314  004737  052352          JSR     PC,$TYPEC       ;;GO TYPE A NULL
10772  052320  105337  052416          DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
10773  052324  000770                  BR      7$              ;;LOOP
10774
10775                                  ;HORIZONTAL TAB PROCESSOR
10776
10777  052326  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
10778  052332  004737  052352          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
10779  052336  132737  000007  052416          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
10780  052344  001372                  BNE     9$              ;;TAB STOP
10781  052346  005726                  TST     (SP)+           ;;POP SPACE OFF STACK
10782  052350  000724                  BR      2$              ;;GET NEXT CHARACTER
10783  052352  105777  126572          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
10784  052356  100375                  BPL     $TYPEC
10785  052360  116677  000002  126564          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
10786  052366  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
10787  052374  001003                  BNE     1$              ;;BRANCH IF NO
10788  052376  105037  052416          CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
10789  052402  000406                  BR      $TYPEX          ;;EXIT
10790  052404  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
10791  052412  001402                  BEQ     $TYPEX          ;;BRANCH IF YES
10792  052414  105227                  INCB    (PC)+           ;;COUNT THE CHARACTER
10793  052416  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
10794  052420  000207                  $TYPEX: RTS     PC
10795
10796
10797
10798
10799                                  ;;*****************************************************************
10800                                  .SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
10801                                  ;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
10802                                  ;*      USES ALL REGISTERS (R0-R5)
10803                                  ;*
10804                                  ;*      ENTER WITH   JSR   PC,M.DPIM
10805                                  ;*                   MULTIPLIER IN R2-R3
10806                                  ;*                   MULTIPLICAND IN R4-R5
10807                                  ;*                   PRODUCT RETURNED IN R0-R1-R2-R3
10808                                  ;*
10809                                  ;;*****************************************************************
10810  052422  005000                  M.DPIM: CLR     R0              ;CLEAR HI ORDER WORDS
10811  052424  005001                          CLR     R1
10812  052426  012746  000041                  MOV     #41,-(SP)       ;MOVE 33 (DEC) TO COUNTER
10813  052432  006000                  M.DP01: ROR     R0
10814  052434  006001                          ROR     R1
10815  052436  006002                          ROR     R2              ;SHIFT TO ADD
10816  052440  006003                          ROR     R3
10817  052442  103003                          BCC     M.DP02          ;NO CARRY NO ADD
10818  052444  060501                          ADD     R5,R1
10819  052446  005500                          ADC     R0              ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
10820  052450  060400                          ADD     R4,R0           ;  PRODUCT
10821  052452  005316                  M.DP02: DEC     @SP             ;DECREMENT COUNTER
10822  052454  001366                          BNE     M.DP01
10823  052456  005726                          TST     (SP)+           ;REMOVE THE COUNTER
```

```
10824  052460  000207                          RTS     PC
10825
10826
10827                                  ;;*****************************************************************
10828                                  .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
10829                                  ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
10830                                  ;*              USES ALL REGISTERS (R0-R5)
10831                                  ;*
10832                                  ;*              ENTER WITH  JSR   PC,M.DPID
10833                                  ;*                      DIVIDEND IN R0-R1-R2-R3
10834                                  ;*                      DIVISOR IN R4-R5
10835                                  ;*                      REMAINDER RETURNED IN R0-R1
10836                                  ;*                      QUOTIENT RETURNED IN R2-R3
10837                                  ;*
10838                                  ;;*****************************************************************
10839  052462  012746  000040         M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
10840  052466  010446                          MOV     R4,-(SP)        ;HI ORDER
10841  052470  010546                          MOV     R5,-(SP)        ;LO ORDER DIVISOR TO THE STACK
10842  052472  005466  000002                 NEG     2(SP)           ;FORM NEGATIVE
10843  052476  005416                          NEG     @SP             ;   VERSION OF THE DIVISOR
10844  052500  005666  000002                 SBC     2(SP)
10845  052504  061601                          ADD     @SP,R1
10846  052506  005500                          ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
10847  052510  066600  000002                 ADD     2(SP),R0
10848  052514  103445                          BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
10849  052516  005046                          CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
10850  052520  006103                  M.DP40: ROL     R3
10851  052522  006102                          ROL     R2
10852  052524  006101                          ROL     R1
10853  052526  006100                          ROL     R0
10854  052530  005716                          TST     @SP             ;TEST "CARRY INDICATOR"
10855  052532  001410                          BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
10856  052534  005016                          CLR     @SP             ;CLEAR UP FOR NEXT TIME
10857  052536  066601  000002                 ADD     2(SP),R1
10858  052542  005500                          ADC     R0              ;ADD -(DIVISOR)
10859  052544  005516                          ADC     @SP     ; I     ;SET "CARRY"
10860  052546  066600  000004                 ADD     4(SP),R0 ;<
10861  052552  000404                          BR      M.DP42
10862  052554  060501                  M.DP41: ADD     R5,R1
10863  052556  005500                          ADC     R0              ;ADD +(DIVISOR)
10864  052560  005516                          ADC     @SP     ; I     ;SET "CARRY"
10865  052562  060400                          ADD     R4,R0   ;<
10866  052564  005516                  M.DP42: ADC     @SP             ;SET "CARRY"
10867  052566  005716                          TST     @SP             ;TEST THE UPDATE INDICATOR
10868  052570  001401                          BEQ     .+4     ;>      ;IF ZERO FORGET IT
10869  052572  005203                          INC     R3      ; I     ;NO CARRY POSSIBLE HERE
10870  052574  005366  000006                 DEC     6(SP)   ;<      ;DECREMENT COUNTER
10871  052600  003347                          BGT     M.DP40          ;BR IF MORE TO DO
10872  052602  006003                          ROR     R3
10873  052604  103404                          BCS     M.DP44
10874  052606  060501                          ADD     R5,R1
10875  052610  005500                          ADC     R0
10876  052612  060400                          ADD     R4,R0
10877  052614  000241                          CLC
10878  052616  006103                  M.DP44: ROL     R3
10879  052620  062706  000010                 ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
```

```
10880  052624  000242                          CLV
10881  052626  000207                          RTS     PC
10882  052630  062706  000006        M.DP50: ADD     #6.SP
10883  052634  000262                          SEV
10884  052636  000207                          RTS     PC
10885
10886
10887
10888                                  .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
10889
10890                                  ;;**********************************************************************
10891                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
10892                                  ;*UNSIGNED OCTAL ASCIZ NUMBER.
10893                                  ;*CALL
10894                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
10895                                  ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
10896                                  ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
10897
10898
10899  052640  104407                  $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
10900  052642  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
10901  052646  012705  052757                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
10902  052652  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
10903  052656  012703  177770                  MOV     #^C7,R3         ;;MASK
10904  052662  012100                          MOV     (R1)+,R0        ;;LOWER WORD
10905  052664  012101                          MOV     (R1)+,R1        ;;HIGH WORD
10906  052666  005002                          CLR     R2              ;;TERMINATOR
10907  052670  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
10908  052672  010002                          MOV     R0,R2           ;;GET THIS DIGIT
10909  052674  005304                          DEC     R4              ;;COUNT THIS CHARACTER
10910  052676  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
10911  052700  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
10912  052702  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
10913  052704  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
10914  052710  104410                          RESREG                  ;;RESTORE ALL REGISTERS
10915  052712  000207                          RTS     PC              ;;RETURN TO USER
10916  052714  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
10917  052716  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
10918  052720  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
10919  052722  006001                          ROR     R1
10920  052724  006000                          ROR     R0
10921  052726  006001                          ROR     R1
10922  052730  006000                          ROR     R0
10923  052732  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
10924  052734  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
10925  052740  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
10926  052742  000016                  $OCTVL: .BLKB   14.              ;;RESERVE DATA TABLE
10927                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
10928
10929                                  ;;**********************************************************************
10930                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
10931                                  ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
10932                                  ;*POSITIVE.
10933                                  ;*CALL
10934                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
10935                                  ;*      JSR     PC,@#$DB2D
```

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


052760  104407                    $DB2D:  SAVREG                     ;;SAVE REGISTERS
052762  016602  000002                    MOV     2(SP),R2           ;;PICKUP THE DATA POINTER
052766  012700  053140                    MOV     #$DECVL,R0         ;;GET ADDRESS OF "$DECVL" STRING
052772  010066  000002                    MOV     R0,2(SP)           ;;PUT ADDRESS OF ASCIZ STRING ON STACK
052776  012201                            MOV     (R2)+,R1           ;;PICKUP THE BINARY NUMBER
053000  012202                            MOV     (R2)+,R2
053002  012737  000012  053056            MOV     #10.,4$            ;;SET UP TO DO 10 CONVERSIONS
053010  012704  053070                    MOV     #$TNPWR,R4         ;;ADDRESS OF TEN POWER
053014  012705  053072                    MOV     #$TNPWR+2,R5
053020  005003                    1$:     CLR     R3                 ;;CLEAR PARTIAL
053022  161401                    2$:     SUB     (R4),R1            ;;SUBTRACT TEN POWER
053024  005602                            SBC     R2
053026  161502                            SUB     (R5),R2
053030  002402                            BLT     3$                 ;;BR IF TEN POWER TO LARGE
053032  005203                            INC     R3                 ;;ADD 1 TO PARTIAL
053034  000772                            BR      2$                 ;;LOOP
053036  062401                    3$:     ADD     (R4)+,R1           ;;RESTORE SUBTRACTED VALUE
053040  005502                            ADC     R2
053042  062402                            ADD     (R4)+,R2
053044  022525                            CMP     (R5)+,(R5)+        ;;MOVE TO NEXT TEN POWER
053046  052703  000060                    BIS     #'0,R3             ;;CHANGE PARTIAL TO ASCII
053052  110320                            MOVB    R3,(R0)+           ;;SAVE IT
053054  005327                            DEC     (PC)+              ;;DONE?
053056  000000                    4$:     .WORD   0
053060  001357                            BNE     1$                 ;;BR IF NO
053062  105020                            CLRB    (R0)+              ;;TERMINATOR
053064  104410                            RESREG                     ;;RESTORE REGISTERS
053066  000207                            RTS     PC                 ;;RETURN
053070  145000                    $TNPWR: 145000                     ;;1.0E09
053072  035632                            35632
053074  160400                            160400                     ;;1.0E08
053076  002765                            2765
053100  113200                            113200                     ;;1.0E07
053102  000230                            230
053104  041100                            041100                     ;;1.0E06
053106  000017                            17
053110  103240                            103240                     ;;1.0E05
053112  000001                            1
053114  023420                            23420                      ;;1.0E04
053116  000000                            0
053120  001750                            1750                       ;;1.0E03
053122  000000                            0
053124  000144                            144                        ;;1.0E02
053126  000000                            0
053130  000012                            12                         ;;1.0E01
053132  000000                            0
053134  000001                            1                          ;;1.0E00
053136  000000                            0
053140  000014                    $DECVL: .BLKB   12.                ;;RESERVE STORAGE FOR ASCIZ STRING
                                  .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

```
                        ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
                        ;*LEADING NUMBERS.
                        ;*CALL
                        ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
                        ;*      JSR     PC,@#$SUPRS
                        
053154  010046          $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
053156  016600  000004          MOV     4(SP),R0        ;;PICKUP THE POINTER
053162  105710          1$:     TSTB    (R0)            ;;TERMINATOR?
053164  001403                  BEQ     2$              ;;BR IF YES
053166  122720  000060          CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
053172  001773                  BEQ     1$              ;;BR IF YES
053174  005300          2$:     DEC     R0              ;;BACKUP BY "1"
053176  010037  053204          MOV     R0,3$           ;;SAVE FOR TYPING
053202  104401                  TYPE                    ;;GO TYPE
053204  000000          3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
053206  012600                  MOV     (SP)+,R0        ;;RESTORE R0
053210  012616                  MOV     (SP)+,(SP)      ;;RESTORE THE STACK
053212  000207                  RTS     PC              ;;RETURN
                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                        
                        ;;**************************************************************
                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                        ;*      .BYTE   M               ;;M=1 OR 0
                        ;*                                      ;;1=TYPE LEADING ZEROS
                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                        ;*
                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                        ;*$TYPOS OR $TYPOC
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                        ;*
                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                        ;*CALL:
                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                        
053214  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
053220  116637  000001  053437  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
053226  112637  053441          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
053232  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
053236  000406                  BR      $TYPON
053240  112737  000001  053437  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
053246  112737  000006  053441          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
053254  112737  000005  053436  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
053262  010346                  MOV     R3,-(SP)        ;;SAVE R3
053264  010446                  MOV     R4,-(SP)        ;;SAVE R4
053266  010546                  MOV     R5,-(SP)        ;;SAVE R5
```

```
053270  113704  053441                   MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
053274  005404                           NEG     R4
053276  062704  000006                   ADD     #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
053302  110437  053440                   MOVB    R4,$OMODE        ;;SAVE IT FOR USE
053306  113704  053437                   MOVB    $OFILL,R4        ;;GET THE ZERO FILL SWITCH
053312  016605  000012                   MOV     12(SP),R5        ;;PICKUP THE INPUT NUMBER
053316  005003                           CLR     R3               ;;CLEAR THE OUTPUT WORD
053320  006105                   1$:     ROL     R5               ;;ROTATE MSB INTO "C"
053322  000404                           BR      3$               ;;GO DO MSB
053324  006105                   2$:     ROL     R5               ;;FORM THIS DIGIT
053326  006105                           ROL     R5
053330  006105                           ROL     R5
053332  010503                           MOV     R5,R3
053334  006103                   3$:     ROL     R3               ;;GET LSB OF THIS DIGIT
053336  105337  053440                   DECB    $OMODE           ;;TYPE THIS DIGIT?
053342  100016                           BPL     7$               ;;BR IF NO
053344  042703  177770                   BIC     #177770,R3       ;;GET RID OF JUNK
053350  001002                           BNE     4$               ;;TEST FOR 0
053352  005704                           TST     R4               ;;SUPPRESS THIS 0?
053354  001403                           BEQ     5$               ;;BR IF YES
053356  005204                   4$:     INC     R4               ;;DON'T SUPPRESS ANYMORE 0'S
053360  052703  000060                   BIS     #'0,R3           ;;MAKE THIS DIGIT ASCII
053364  052703  000040           5$:     BIS     #' ,R3           ;;MAKE ASCII IF NOT ALREADY
053370  110337  053434                   MOVB    R3,8$            ;;SAVE FOR TYPING
053374  104401  053434                   TYPE    ,8$              ;;GO TYPE THIS DIGIT
053400  105337  053436           7$:     DECB    $OCNT            ;;COUNT BY 1
053404  003347                           BGT     2$               ;;BR IF MORE TO DO
053406  002402                           BLT     6$               ;;BR IF DONE
053410  005204                           INC     R4               ;;INSURE LAST DIGIT ISN'T A BLANK
053412  000744                           BR      2$               ;;GO DO THE LAST DIGIT
053414  012605                   6$:     MOV     (SP)+,R5         ;;RESTORE R5
053416  012604                           MOV     (SP)+,R4         ;;RESTORE R4
053420  012603                           MOV     (SP)+,R3         ;;RESTORE R3
053422  016666  000002  000004           MOV     2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
053430  012616                           MOV     (SP)+,(SP)
053432  000002                           RTI                      ;;RETURN
053434     000                   8$:     .BYTE   0                ;;STORAGE FOR ASCII DIGIT
053435     000                           .BYTE   0                ;;TERMINATOR FOR TYPE ROUTINE
053436     000                   $OCNT:  .BYTE   0                ;;OCTAL DIGIT COUNTER
053437     000                   $OFILL: .BYTE   0                ;;ZERO FILL SWITCH
053440  000000                   $OMODE: .WORD   0                ;;NUMBER OF DIGITS TO TYPE
                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

        053442                   $TYPDS:
053442  010046                           MOV     R0,-(SP)         ;;PUSH R0 ON STACK
053444  010146                           MOV     R1,-(SP)         ;;PUSH R1 ON STACK
```

```
111104  053446  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
111105  053450  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
111106  053452  010546                      MOV     R5,-(SP)        ;;PUSH R5 ON STACK
111107  053454  012746  020200              MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
111110  053460  016605  000020              MOV     20(SP),R5       ;;GET THE INPUT NUMBER
111111  053464  100004                      BPL     1$              ;;BR IF INPUT IS POS.
111112  053466  005405                      NEG     R5              ;;MAKE THE BINARY NUMBER POS.
111113  053470  112766  000055  000001      MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
111114  053476  005003          1$:         CLR     R0              ;;ZERO THE CONSTANTS INDEX
111115  053500  012703  053656              MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
111116  053504  112723  000040              MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
111117  053510  005002          2$:         CLR     R2              ;;CLEAR THE BCD NUMBER
111120  053512  016001  053646              MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
111121  053516  160105          3$:         SUB     R1,R5           ;;FORM THIS BCD DIGIT
111122  053520  002402                      BLT     4$              ;;BR IF DONE
111123  053522  005202                      INC     R2              ;;INCREASE THE BCD DIGIT BY 1
111124  053524  000774                      BR      3$
111125  053526  060105          4$:         ADD     R1,R5           ;;ADD BACK THE CONSTANT
111126  053530  005702                      TST     R2              ;;CHECK IF BCD DIGIT=0
111127  053532  001002                      BNE     5$              ;;FALL THROUGH IF 0
111130  053534  105716                      TSTB    (SP)            ;;STILL DOING LEADING 0'S?
111131  053536  100407                      BMI     7$              ;;BR IF YES
111132  053540  106316          5$:         ASLB    (SP)            ;;MSD?
111133  053542  103003                      BCC     6$              ;;BR IF NO
111134  053544  116663  000001  177777      MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
111135  053552  052702  000060          6$: BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
111136  053556  052702  000040          7$: BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
111137  053562  110223                      MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
111140  053564  005720                      TST     (R0)+           ;;JUST INCREMENTING
111141  053566  020027  000010              CMP     R0,#10          ;;CHECK THE TABLE INDEX
111142  053572  002746                      BLT     2$              ;;GO DO THE NEXT DIGIT
111143  053574  003002                      BGT     8$              ;;GO TO EXIT
111144  053576  010502                      MOV     R5,R2           ;;GET THE LSD
111145  053600  000764                      BR      6$              ;;GO CHANGE TO ASCII
111146  053602  105726          8$:         TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
111147  053604  100003                      BPL     9$              ;;BR IF NO
111150  053606  116663  177777  177776      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
111151  053614  105013          9$:         CLRB    (R3)            ;;SET THE TERMINATOR
111152  053616  012605                      MOV     (SP)+,R5        ;;POP STACK INTO R5
111153  053620  012603                      MOV     (SP)+,R3        ;;POP STACK INTO R3
111154  053622  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
111155  053624  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
111156  053626  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
111157  053630  104401  053656              TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
111160  053634  016666  000002  000004      MOV     2(SP),4(SP)     ;;ADJUST THE STACK
111161  053642  012616                      MOV     (SP)+,(SP)
111162  053644  000002                      RTI                     ;;RETURN TO USER
111163  053646  023420          $DTBL:      10000.
111164  053650  001750                      1000.
111165  053652  000144                      100.
111166  053654  000012                      10.
111167  053656  000004          $DBLK:      .BLKW   4
111170                          .SBTTL  ERROR HANDLER ROUTINE
111171
111172                          ;;**********************************************************************
111173                          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

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

053666                                  $ERROR:
053666  105237  001103                  7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
053672  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
053674  013777  001102  125240          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
053702  032777  002000  125230          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
053710  001402                          BEQ     1$              ;;NO - SKIP
053712  104401  001310                  TYPE    ,$BELL          ;;RING BELL
053716  005237  001112                  1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
053722  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
053726  162737  000002  001116          SUB     #2,$ERRPC
053734  117737  125156  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
053742  032777  020000  125170          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
053750  001004                          BNE     20$             ;;SKIP TYPEOUTS
053752  004737  040166                  JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
053756  104401  001315                  TYPE    ,$CRLF
053762                                  20$:
053762  122737  000001  001340          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
053770  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
053772  113737  001114  054004          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
054000  004737  054646                  JSR     PC,$APY4        ;;REPORT FATAL ERROR TO APT
054004  000                             21$:    .BYTE   0
054005  000                             .BYTE   0
054006  000777                          22$:    BR      22$             ;;APT ERROR LOOP
054010  005777  125124                  2$:     TST     @SWR            ;;HALT ON ERROR
054014  100001                          BPL     3$              ;;SKIP IF CONTINUE
054016  000000                          HALT                    ;;HALT ON ERROR!
054020  032777  001000  125112          3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
054026  001402                          BEQ     4$              ;;BR IF NO
054030  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
054034  005737  001306                  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
054040  001402                          BEQ     5$              ;;BR IF NONE
054042  013716  001306                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
054046                                  5$:
054046  022737  025056  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
054054  001001                          BNE     6$              ;;BRANCH IF NO
054056  000000                          HALT                    ;;YES
054060                                  6$:
054060  000002                          RTI                     ;;RETURN
                                        .SBTTL  TTY INPUT ROUTINE

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

                                        .DSABL  LSB
```

```
11216                                   ;;*****************************************************************
11217                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11218                                   ;*CALL:
11219                                   ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
11220                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
11221                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
11222                                   ;
11223
11224  054362  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
11225  054364  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
11226  054372  105777  125046          1$:     TSTB    @$TKS           ;;WAIT FOR
11227  054376  100375                          BPL     1$              ;;A CHARACTER
11228  054400  117766  125042  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
11229  054406  042766  177600  000004          BIC     #^C177,4(SP)    ;;GET RID OF JUNK IF ANY
11230  054414  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
11231  054422  001013                          BNE     3$              ;;BRANCH IF NO
11232  054424  105777  125014          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
11233  054430  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
11234  054432  117746  125010                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
11235  054436  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
11236  054442  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
11237  054446  001366                          BNE     2$              ;;IF NOT DISCARD IT
11238  054450  000750                          BR      1$              ;;YES, RESUME
11239  054452  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
11240  054460  002407                          BLT     4$              ;;BRANCH IF YES
11241  054462  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
11242  054470  003003                          BGT     4$              ;;BRANCH IF YES
11243  054472  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
11244  054500  000002                  4$:     RTI                     ;;GO BACK TO USER
11245  054502  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
11246  054507     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
11247  054514  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR =
11248  054522  020075     000
11249  054525     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
11250  054532  036440  000040
11251
11252
11253
11254                                   .SBTTL  POWER DOWN AND UP ROUTINES
11255
11256                                   ;POWER DOWN ROUTINE
11257  054536  012737  054250  000024  $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
11258  054544  000000                          HALT                    ;HANG UP
11259  054546  000776                          BR      .-2
11260
11261                                   ;POWER UP ROUTINE
11262  054550  005037  054322          $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
11263  054554  005237  054322          4$:     INC     $PWRCT
11264  054560  001375                          BNE     4$
11265  054562  012737  054236  000024          MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
11266  054570  012737  000340  000026          MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11267  054576  012737  000340  000036          MOV     #PR7,TRAPVEC+2
11268  054604  012706  001100                  MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
11269  054610  104401  054324                  TYPE    ,PWRMSG         ;TYPE "POWER FAILED"
11270  054614  000005                          RESET                   ;CLEAR THE UNIBUS
11271  054616  000177  124564                  JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
11272  054322  000000                  $PWRCT: .WORD   0
11273  054324  035015  047520  042527  PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
11274  054332  020122  040506  046111
11275  054340  042105  005015     000
11276          054346                          .EVEN
11277
11278
11279
11280                                  ;;**********************************************************************
11281                                  ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
11282                                  ;*CALLED BY "SCOPER"
11283                                  ;;**********************************************************************
11284  054346  105737  001103          $SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
11285  054352  001406                           BEQ     6$              ;BR IF NOT
11286  054354  032777  001000  124556           BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
11287  054362  001402                           BEQ     6$              ;BR IF NOT
11288  054364  013716  001110                   MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
11289  054370  000002                  6$:      RTI                     ;RETURN
11290
11291
11292
11293                                  .SBTTL  SCOPE HANDLER ROUTINE
11294
11295                                  ;;**********************************************************************
11296                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11297                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11298                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11299                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11300                                  ;*SW14=1         LOOP ON TEST
11301                                  ;*SW11=1         INHIBIT ITERATIONS
11302                                  ;*SW09=1         LOOP ON ERROR
11303                                  ;*CALL
11304                                  ;*      SCOPE           ;;SCOPE=IOT
11305
11306  054372                          $SCOPE:
11307  054372  005737  001304                  TST     $TIMES          ;CHECK CURRENT ITERATION NUMBER
11308  054376  001404                           BEQ     $XTSTR          ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
11309  054400  032777  040000  124532  1$:      BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11310  054406  001101                           BNE     $OVER           ;;YES IF SW14=1
11311                                  ;*****START OF CODE FOR THE XOR TESTER*****
11312  054410  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11313                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11314  054412  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11315  054416  012737  054436  000004           MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11316  054424  005737  177060                   TST     @#177060        ;;TIME OUT ON XOR?
11317  054430  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11318  054434  000450                           BR      $SVLAD          ;;GO TO THE NEXT TEST
11319  054436  022626                  5$:      CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11320  054440  012637  000004                   MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11321  054444  000413                           BR      7$              ;;LOOP ON THE PRESENT TEST
11322  054446                          6$:;*****END OF CODE FOR THE XOR TESTER*****
11323  054446  105737  001103          2$:      TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11324  054452  001421                           BEQ     3$              ;;BR IF NO
11325  054454  123737  001115  001103           CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
11326  054462  101015                           BHI     3$              ;;BR IF NO
11327  054464  032777  001000  124446           BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
```

```
11328  054472  001404                          BEQ     4$                ;;BR IF NO
11329  054474  013737  001110  001106  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
11330  054502  000443                          BR      $OVER
11331  054504  105037  001103          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
11332  054510  005037  001304                  CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11333  054514  000412                          BR      1$                ;;ESCAPE TO THE NEXT TEST
11334  054516  032777  004000  124414  3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
11335  054524  001006                          BNE     1$                ;;BR IF YES
11336  054526  005237  001104                  INC     $ICNT             ;;INCREMENT ITERATION COUNT
11337  054532  023737  001304  001104          CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
11338  054540  002024                          BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
11339  054542  012737  000001  001104  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
11340  054550  013737  054626  001304          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
11341  054556  105237  001102          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
11342  054562  113737  001102  001324          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
11343  054570  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
11344  054574  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
11345  054600  005037  001306                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11346  054604  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11347  054612  013777  001102  124322  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
11348  054620  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
11349  054624  000002                          RTI                       ;;FIXES PS
11350  054626  003720                  $MXCNT: 2000.                     ;;MAX. NUMBER OF ITERATIONS
11351                                  .SBTTL  APT COMMUNICATIONS ROUTINE
11352
11353                                  ;;**************************************************************
11354  054630  112737  000001  055074  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
11355  054636  112737  000001  055072  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
11356  054644  000403                          BR      $ATYC
11357  054646  112737  000001  055074  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
11358  054654                          $ATYC:
11359  054654  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
11360  054656  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
11361  054660  105737  055072                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
11362  054664  001450                          BEQ     5$                ;;IF NOT:  BR
11363  054666  122737  000001  001340          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
11364  054674  001031                          BNE     3$                ;;IF NOT:  BR
11365  054676  132737  000100  001341          BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
11366  054704  001425                          BEQ     3$                ;;IF NOT:  BR
11367  054706  017600  000004                  MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
11368  054712  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
11369  054720  005737  001320          1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
11370  054724  001375                          BNE     1$                ;;IF NOT:  WAIT
11371  054726  010037  001334                  MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
11372  054732  105720                  2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
11373  054734  001376                          BNE     2$
11374  054736  163700  001334                  SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
11375  054742  006200                          ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
11376  054744  010037  001336                  MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
11377  054750  012737  000004  001320          MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
11378  054756  000413                          BR      5$
11379  054760  017637  000004  055004  3$:     MOV     @4(SP),.4$        ;;PUT MSG ADDR IN JSR LINKAGE
11380  054766  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
11381  054774  013746  177776                  MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
11382  055000  004737  052140                  JSR     PC,$TYPE          ;;CALL TYPE MACRO
11383  055004  000000                  4$:     .WORD   0
```

```
11384  055006                          5$:
11385  055006  105737  055074          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
11386  055012  001416                          BEQ     12$             ;;IF NOT:  BR
11387  055014  005737  001340                  TST     $ENV            ;;RUNNING UNDER APT?
11388  055020  001413                          BEQ     12$             ;;IF NOT:  BR
11389  055022  005737  001320          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
11390  055026  001375                          BNE     11$             ;;IF NOT:  WAIT
11391  055030  017637  000004  001322          MOV     @4(SP),$FATAL   ;;GET ERROR #
11392  055036  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
11393  055044  005237  001320                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
11394  055050  105037  055074          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
11395  055054  105037  055073                  CLRB    $LFLG           ;;CLEAR LOG FLAG
11396  055060  105037  055072                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
11397  055064  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11398  055066  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11399  055070  000207                          RTS     PC              ;;RETURN
11400  055072     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
11401  055073     000          $LFLG:  .BYTE   0               ;;LOG FLAG
11402  055074     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
11403          055076                  .EVEN
11404          000200          APTSIZE=200
11405          000001          APTENV=001
11406          000100          APTSPOOL=100
11407          000040          APTCSUP=040
11408                          .SBTTL  ROUTINE TO SIZE MEMORY
11409
11410                          ;;**************************************************************
11411                          ;*CALL:
11412                          ;*      JSR     PC,$SIZE
11413                          ;*      RETURN
11414                          ;*$LSTAD WILL CONTAIN:
11415                          ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11416                          ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11417                          ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11418                          ;*$KT11 IS THE MEMORY MANAGEMENT KEY
11419                          ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11420                          ;*      MUST BE SETUP BEFORE THE CALL
11421                          ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11422                          ;*      DETERMINED BY ROUTINE
11423
11424  055076  010046          $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
11425  055100  010146                  MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11426  055102  010246                  MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
11427  055104  010346                  MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
11428  055106  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
11429  055112  013746  000006          MOV     @#ERRVEC+2,-(SP)
11430  055116  010600                  MOV     SP,R0           ;;SAVE THE STACK POINTER
11431                          ;;SET THE ERRVEC PS TO THE PRESENT PS
11432  055120  104400                  TRAP                    ;;PUSH OLD PSW AND PC ON STACK
11433  055122  012637  000006          MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
11434  055126  012701  003776          MOV     #3776,R1        ;;SETUP ADDRESS
11435  055132  105727                  TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
11436  055134  000200          $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
11437  055136  100062                  BPL     $CORE           ;;BR IF NO
11438  055140  012737  055276  000004  MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
11439  055146  005737  177572          TST     @#SR0           ;;KT11 ARE YOU THERE?
```

```
11440  055152  052737  100000  055134          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
11441  055160  005046                          CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
11442  055162  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
11443  055166  012703  000010                  MOV     #10,R3          ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
11444  055172  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
11445  055200  011622                          MOV     (SP),(R2)+      ;;LOAD "PAR"
11446  055202  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
11447  055206  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
11448  055210  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
11449  055214  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
11450  055216  012737  055234  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
11451  055224  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
11452  055232  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
11453  055234  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
11454  055236  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
11455  055242  012737  055266  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
11456  055250  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
11457  055254  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
11458  055260  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
11459  055264  101371                          BHI     4$              ;;NO--TRY IT
11460  055266  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
11461  055270  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
11462  055274  000421                          BR      $SIZEX
11463  055276  042737  100000  055134  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
11464  055304  012737  055334  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
11465  055312  005002                          CLR     R2              ;;SET UP BANK
11466  055314  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
11467  055320  062702  000040                  ADD     #40,R2          ;;1K STEP
11468  055324  005711                          TST     (R1)            ;;TRAP ON TIME OUT
11469  055326  022701  177776                  CMP     #177776,R1      ;;LAST ONE
11470  055332  001370                          BNE     1$              ;;NO--TRY AGAIN
11471  055334  162701  004000          $CROUT: SUB     #4000,R1
11472  055340  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
11473  055344  010006                          MOV     R0,SP           ;;RESTORE THE STACK
11474  055346  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
11475  055352  012637  000004                  MOV     (SP)+,@#ERRVEC
11476  055356  010137  055400                  MOV     R1,$LSTAD       ;;LAST ADDRESS
11477  055362  010237  055402                  MOV     R2,$LSTBK       ;;LAST BANK
11478  055366  012603                          MOV     (SP)+,R3        ;;RESTORE R3
11479  055370  012602                          MOV     (SP)+,R2        ;;RESTORE R2
11480  055372  012601                          MOV     (SP)+,R1        ;;RESTORE R1
11481  055374  012600                          MOV     (SP)+,R0        ;;RESTORE R0
11482  055376  000207                          RTS     PC
11483  055400  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
11484  055402  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
11485                           .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11486
11487                           ;;******************************************************************
11488                           ;*SAVE R0-R5
11489                           ;*CALL:
11490                           ;*      SAVREG
11491                           ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11492                           ;*
11493                           ;*TOP---(+16)
11494                           ;* +2---(+18)
11495                           ;* +4---R5
```

```
11496                                   ;* +6---R4
11497                                   ;* +8---R3
11498                                   ;*+10---R2
11499                                   ;*+12---R1
11500                                   ;*+14---R0
11501
11502  055404                           $SAVREG:
11503  055404  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11504  055406  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11505  055410  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11506  055412  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11507  055414  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11508  055416  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11509  055420  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11510  055424  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11511  055430  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11512  055434  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11513  055440  000002                          RTI
11514
11515                                   ;*RESTORE R0-R5
11516                                   ;*CALL:
11517                                   ;*      RESREG
11518  055442                           $RESREG:
11519  055442  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11520  055446  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11521  055452  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11522  055456  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11523  055462  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11524  055464  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11525  055466  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11526  055470  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11527  055472  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11528  055474  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11529  055476  000002                          RTI
11530                                   .SBTTL  TRAP DECODER
11531
11532                                   ;;*****************************************************************
11533                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11534                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11535                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11536                                   ;*GO TO THAT ROUTINE.
11537
11538  055500  010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
11539  055502  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
11540  055506  005740                          TST     -(R0)           ;;BACKUP BY 2
11541  055510  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
11542  055512  006300                          ASL     R0              ;;POSITION FOR INDEXING
11543  055514  016000  055534                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
11544  055520  000200                          RTS     R0              ;;GO TO ROUTINE
11545
11546
11547                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
11548
11549  055522  011646                   $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
11550  055524  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
11551  055532  000002                          RTI                     ;;RESTORE THE PSW
```

```
11552
11553                                   .SBTTL  TRAP TABLE
11554
11555                                   ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
11556                                   ;*BY THE "TRAP" INSTRUCTION.
11557
11558                                   ;       ROUTINE
11559                                   ;       -------
11560  055534  055522                  $TRPAD: .WORD   $TRAP2
11561  055536  052140                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
11562  055540  053240                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
11563  055542  053214                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
11564  055544  053254                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
11565  055546  053442                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
11566
11567
11568  055550  054062                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
11569  055552  055404                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
11570  055554  055442                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
11571  055556  054346                          SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
11572
11573  055560  020040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
11574  055566  052123  000040
11575  055572  025052  020040     000  AS2SP2: .ASCIZ  /**  /
11576  055577     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PARITY ERROR/
11577  055604  020123  040520  044522
11578  055612  054524  042440  051122
11579  055620  051117     000
11580  055623     116  047117  042455  EM2:    .ASCIZ  /NON-EXISTANT MEMORY/
11581  055630  044530  052123  047101
11582  055636  020124  042515  047515
11583  055644  054522     000
11584  055647     116  047117  042455  EM3:    .ASCIZ  /NON-EXISTANT DRIVE/
11585  055654  044530  052123  047101
11586  055662  020124  051104  053111
11587  055670  000105
11588  055672  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERROR/
11589  055700  042511  042114  042440
11590  055706  051122  051117     000
11591  055713     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
11592  055720  020123  044524  042515
11593  055726  052517  000124
11594  055732  020104  047524  041440  EM6:    .ASCIZ  /D TO C PARITY ERROR/
11595  055740  050040  051101  052111
11596  055746  020131  051105  047522
11597  055754  000122
11598  055756  051104  053111  020105  EM7:    .ASCIZ  /DRIVE DETECTED PARITY ERROR/
11599  055764  042504  042524  052103
11600  055772  042105  050040  051101
11601  056000  052111  020131  051105.
11602  056006  047522  000122
11603  056012  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
11604  056020     000
11605  056021     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
11606  056026  046040  051517  000123
11607  056034  046111  042514  040507  EM12:   .ASCIZ  /ILLEGAL FUNCTION/
```

```
11608  056042  020114  052506  041516
11609  056050  044524  047117     000
11610  056055     120  047522  051107  EM13:   .ASCIZ  /PROGRAMMING ERROR/
11611  056062  046501  044515  043516
11612  056070  042440  051122  051117
11613  056076     000
11614  056077     116  047117  042455  EM14:   .ASCIZ  /NON-EXISTANT FUNCTION/
11615  056104  044530  052123  047101
11616  056112  020124  052506  041516
11617  056120  044524  047117     000
11618  056125     104  044522  042526  EM15:   .ASCIZ  /DRIVE TYPE ERROR/
11619  056132  052040  050131  020105
11620  056140  051105  047522  000122
11621  056146  047506  046522  052101  EM16:   .ASCIZ  /FORMAT ERROR/
11622  056154  042440  051122  051117
11623  056162     000
11624  056163     127  044522  042524  EM17:   .ASCIZ  /WRITE LOCK ERROR/
11625  056170  046040  041517  020113
11626  056176  051105  047522  000122
11627  056204  051104  053111  020105  EM20:   .ASCIZ  /DRIVE UNSAFE/
11628  056212  047125  040523  042506
11629  056220     000
11630  056221     123  042505  020113  EM21:   .ASCIZ  /SEEK INCOMPLETE/
11631  056226  047111  047503  050115
11632  056234  042514  042524     000
11633  056241     103  046131  047111  EM22:   .ASCIZ  /CYLINDER OVERFLOW/
11634  056246  042504  020122  053117
11635  056254  051105  046106  053517
11636  056262     000
11637  056263     111  046114  043505  EM23:   .ASCIZ  /ILLEGAL CYLINDER ADDRESS/
11638  056270  046101  041440  046131
11639  056276  047111  042504  020122
11640  056304  042101  051104  051505
11641  056312  000123
11642  056314  051104  053111  020105  EM24:   .ASCIZ  /DRIVE OFF TRACK/
11643  056322  043117  020106  051124
11644  056330  041501  000113
11645  056334  051104  053111  020105  EM25:   .ASCIZ  /DRIVE TIMING ERROR/
11646  056342  044524  044515  043516
11647  056350  042440  051122  051117
11648  056356     000
11649  056357     104  052101  020101  EM26:   .ASCIZ  /DATA LATE/
11650  056364  040514  042524     000
11651  056371     103  047117  051124  EM27:   .ASCIZ  /CONTROLLER TIMEOUT/
11652  056376  046117  042514  020122
11653  056404  044524  042515  052517
11654  056412  000124
11655  056414  050117  051105  052101  EM30:   .ASCIZ  /OPERATION INCOMPLETE/
11656  056422  047511  020116  047111
11657  056430  047503  050115  042514
11658  056436  042524     000
11659  056441     110  040505  042504  EM31:   .ASCIZ  /HEADER VRC ERROR/
11660  056446  020122  051126  020103
11661  056454  051105  047522  000122
11662  056462  040504  040524  041440  EM32:   .ASCIZ  /DATA CHECK ERROR/
11663  056470  042510  045503  042440
```

```
 11664  056476  051122  051117     000
 11665  056503     12  044522  042524  EM33:   .ASCIZ  /WRITE CHECK ERROR/
 11666  056510  041440  042510  045503
 11667  056516  042440  051122  051117
 11668  056524     000
 11669  056525     104  052101  020101  EM34:   .ASCIZ  /DATA MISCOMPARE/
 11670  056532  044515  041523  046517
 11671  056540  040520  042522     000
 11672  056545     116  020117  051104  EM35:   .ASCIZ  /NO DRIVE RESPONSE-UFE & NXD/
 11673  056552  053111  020105  042522
 11674  056560  050123  047117  042523
 11675  056566  052455  042506  023040
 11676  056574  047040  042130     000
 11677  056601     104  044522  042526  EM36:   .ASCIZ  /DRIVE ERROR WILL NOT CLEAR/
 11678  056606  042440  051122  051117
 11679  056614  053440  046111  020114
 11680  056622  047516  020124  046103
 11681  056630  040505  000122
 11682  056634  051104  053111  020105  EM37:   .ASCIZ  /DRIVE STATUS CHANGE WILL NOT CLEAR/
 11683  056642  052123  052101  051525
 11684  056650  041440  040510  043516
 11685  056656  020105  044527  046114
 11686  056664  047040  052117  041440
 11687  056672  042514  051101     000
 11688  056677     101  052124  047047  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
 11689  056704  041040  052125  047040
 11690  056712  020117  052123  052101
 11691  056720  051525  041440  040510
 11692  056726  043516  020105  051117
 11693  056734  043040  052501  052114
 11694  056742     000
 11695  056743     101  052124  047047  EM41:   .ASCIZ  /ATT'N BUT DRIVE NOT AVAILABLE/
 11696  056750  041040  052125  042040
 11697  056756  044522  042526  047040
 11698  056764  052117  040440  040526
 11699  056772  046111  041101  042514
 11700  057000     000
 11701  057001     101  052124  047047  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
 11702  057006  053440  042510  020116
 11703  057014  047516  020124  054105
 11704  057022  042520  052103  042105
 11705  057030     000
 11706  057031     105  051122  051117  EM43:   .ASCIZ  /ERROR GATHERING DRIVE STATUS/
 11707  057036  043440  052101  042510
 11708  057044  044522  043516  042040
 11709  057052  044522  042526  051440
 11710  057060  040524  052524  000123
 11711  057066  052515  052114  050111  EM52:   .ASCIZ  /MULTIPLE DRIVE SELECT/
 11712  057074  042514  042040  044522
 11713  057102  042526  051440  046105
 11714  057110  041505  000124
 11715  057114  042510  042101  051105  EM53:   .ASCIZ  /HEADER COMPARE ERROR/
 11716  057122  041440  046517  040520
 11717  057130  042522  042440  051122
 11718  057136  051117     000
 11719  057141     123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
```

```
 11720  057146  020123  044524  042515
 11721  057154  052517  000124
 11722  057160  051105  047522  020122  EM60:   .ASCIZ  /ERROR IN RECAL FOR RECOVERY/
 11723  057166  047111  051040  041505
 11724  057174  046101  043040  051117
 11725  057202  051040  041505  053117
 11726  057210  051105  000131
 11727  057214  051120  043517  040522  EM61:   .ASCIZ  /PROGRAM ABORTING FATAL ERROR IN RETRY/
 11728  057222  020115  041101  051117
 11729  057230  044524  043516  043040
 11730  057236  052101  046101  042440
 11731  057244  051122  051117  044440
 11732  057252  020116  042522  051124
 11733  057260  000131
 11734  057262  054503  044514  042116  EM62:   .ASCIZ  /CYLINDER MISCOMPARE/
 11735  057270  051105  046440  051511
 11736  057276  047503  050115  051101
 11737  057304  000105
 11738  057306  046103  040505  020122  EM63:   .ASCIZ  /CLEAR CONTROLLER DID NOT CLEAR ERROR/
 11739  057314  047503  052116  047522
 11740  057322  046114  051105  042040
 11741  057330  042111  047040  052117
 11742  057336  041440  042514  051101
 11743  057344  042440  051122  051117
 11744  057352     000
 11745  057353     116  020117  052101  EM64:   .ASCIZ  /NO ATT'N IN ATT'N SUMMARY REGISTER/
 11746  057360  023524  020116  047111
 11747  057366  040440  052124  047047
 11748  057374  051440  046525  040515
 11749  057402  054522  051040  043505
 11750  057410  051511  042524  000122
 11751  057416  047125  047523  044514  EM65:   .ASCIZ  /UNSOLICITED ATTENTION/
 11752  057424  044503  042524  020104
 11753  057432  052101  042524  052116
 11754  057440  047511  000116
 11755  057444  047125  054105  042520  EM66:   .ASCIZ  /UNEXPECTED DATA TYPE ERROR
 11756  057452  052103  042105  042040
 11757  057460  052101  020101  054524
 11758  057466  042520  042440  051122
 11759  057474  051117     000
 11760  057477     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
 11761  057504  042040  042111  047040
 11762  057512  052117  051040  051505
 11763  057520  052105  053440  052111
 11764  057526  020110  046103  040505
 11765  057534  000122
 11766  057536  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRIVE ATT'N/
 11767  057544  041440  042514  051101
 11768  057552  042040  042111  047040
 11769  057560  052117  041440  042514
 11770  057566  051101  042040  044522
 11771  057574  042526  040440  052124
 11772  057602  047047     000
 11773  057605     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN UNLOADING HEADER/
 11774  057612  040514  042524  053440
 11775  057620  042510  020116  047125
```

```
11776  057626  047514  042101  047111
11777  057634  020107  042510  042101
11778  057642  051105     000
11779  057645     103  047117  051124  EM72:   .ASCIZ  /CONTROLLER ERROR WHEN DRIVER SERVICING/
11780  057652  046117  042514  020122
11781  057660  051105  047522  020122
11782  057666  044127  047105  042040
11783  057674  044522  042526  020122
11784  057702  042523  053122  041511
11785  057710  047111  000107
11786  057714  051104  053111  020105  EM73:   .ASCIZ  /DRIVE DETECTED PARITY ERROR/
11787  057722  042504  042524  052103
11788  057730  042105  050040  051101
11789  057736  052111  020131  051105
11790  057744  047522  000122
11791  057750  047125  042504  044506  EM74:   .ASCIZ  /UNDEFINED ERROR/
11792  057756  042516  020104  051105
11793  057764  047522  000122
11794  057770  040515  045522  047111  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
11795  057776  020107  044124  051511
11796  060004  051440  041505  047524
11797  060012  020122  040502  000104
11798  060020  040502  020104  040504  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
11799  060026  040524  053040  051105
11800  060034  043111  047047  053440
11801  060042  052111  020110  042522
11802  060050  042101  020056  041505
11803  060056  020103  043117  046040
11804  060064  051501  020124  042522
11805  060072  051124  020131  051511
11806  060100  000072
11807  060102  042522  051124  020131  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
11808  060110  052523  041503  051505
11809  060116  043123  046125     000
11810  060123     122  052105  054522  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
11811  060130  052440  051516  041525
11812  060136  042503  051523  052506
11813  060144  000114
11814  060146  040503  047116  052117  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
11815  060154  043040  047111  020104
11816  060162  020101  040526  044514
11817  060170  020104  042510  042101
11818  060176  051105  044440  020116
11819  060204  051124  041501  020113
11820  060212  052512  052123  051040
11821  060220  040505  000104
11822  060224  040502  020104  042523  EM102:  .ASCIZ  /BAD SECTOR ERROR ON SECTOR NOT LISTED BAD/
11823  060232  052103  051117  042440
11824  060240  051122  051117  047440
11825  060246  020116  042523  052103
11826  060254  051117  047040  052117
11827  060262  046040  051511  042524
11828  060270  020104  040502  000104
11829  060276  044524  042515  026504  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
11830  060304  052517  020124  047117
11831  060312  051040  040505  020104
```

```
 11832  060320  042110  000122
 11833  060324  044524  042515  026504  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
 11834  060332  052517  020124  047117
 11835  060340  051440  042505  000113
 11836  060346  051124  053111  020105  EM105:  .ASCIZ  /DRIVE SIEZED BY OTHER PORT/
 11837  060354  044523  055105  042105
 11838  060362  041040  020131  052117
 11839  060370  042510  020122  047520
 11840  060376  052122     000
 11841  060401     104  052101  020101  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
 11842  060406  044515  041523  050115
 11843  060414  020122  044127  046111
 11844  060422  020105  040502  020111
 11845  060430  042523  000124
 11846  060434  047516  047040  046505  EM107:  .ASCIZ  /NO NEM ERROR WHEN REF'ING LOC 760000/
 11847  060442  042440  051122  051117
 11848  060450  053440  042510  020116
 11849  060456  042522  023506  047111
 11850  060464  020107  047514  020103
 11851  060472  033067  030060  030060
 11852  060500     000
 11853  060501     111  052116  050122  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
 11854  060506  020124  044127  047105
 11855  060514  041440  052116  046122
 11856  060522  020122  047516  020124
 11857  060530  042122  000131
 11858  060534  047516  040440  052124  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
 11859  060542  047047  047440  020116
 11860  060550  042523  045505     000
 11861  060555     104  044522  042526  EM112:  .ASCIZ  /DRIVE'S CYLINDER INCORRECT/
 11862  060562  051447  041440  046131
 11863  060570  047111  042504  020122
 11864  060576  047111  047503  051122
 11865  060604  041505  000124
 11866  060610  041101  051117  026524  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
 11867  060616  041440  047101  052047
 11868  060624  051040  040505  020104
 11869  060632  051532  000106
 11870  060636  052113  030461  043040  EM114:  .ASCIZ  /KT11 FAILURE/
 11871  060644  044501  052514  042522
 11872  060652     000
 11873  060653     115  046505  050040  EM115:  .ASCIZ  /MEM PARITY ERR/
 11874  060660  051101  052111  020131
 11875  060666  051105  000122
 11876  060672  052503  051122  047105  DH100:  .ASCIZ  /CURRENT COMMAND :/
 11877  060700  020124  047503  046515
 11878  060706  047101  020104  000072
 11879  060714  051120  053105  047511  DH105:  .ASCIZ  /PREVIOUS COMMAND :/
 11880  060722  051525  041440  046517
 11881  060730  040515  042116  035040
 11882  060736     000
 11883  060737     122  033113  030461  DH200:  .ASCIZ  /RK611 REGISTERS :/
 11884  060744  051040  043505  051511
 11885  060752  042524  051522  035040
 11886  060760     000
 11887  060761     122  046505  044501  DH500:  .ASCIZ  /REMAINING REGISTERS NOT VALID/
```

```
 11888  060766  044516  043516  051040
 11889  060774  043505  051511  042524
 11890  061002  051522  047040  052117
 11891  061010  053040  046101  042111
 11892  061016     000
 11893  061017     124  042510  043040  DH501:  .ASCIZ  /THE FOLLOWING REGISTER DATA MAY BE INCORRECT :/
 11894  061024  046117  047514  044527
 11895  061032  043516  051040  043505
 11896  061040  051511  042524  020122
 11897  061046  040504  040524  046440
 11898  061054  054501  041040  020105
 11899  061062  047111  047503  051122
 11900  061070  041505  020124  000072
 11901  061076  051105  020122  041520  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR : TRACK   SECTOR  WD CNT/
 11902  061124  020040  051104  053111
 11903  061112  020105  020040  046503
 11904  061120  042116  020040  020040
 11905  061126  054503  047114  051104
 11906  061134  020040  051124  041501
 11907  061142  020113  020040  042523
 11908  061150  052103  051117  020040
 11909  061156  042127  041440  052116
 11910  061164     000
 11911  061165     110  020111  040502  DH102:  .ASCIZ  /HI BA   LO BA/
 11912  061172  020040  046040  020117
 11913  061200  040502     000
 11914  061203     120  042522  027126  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
 11915  061210  052440  044516  052502
 11916  061216  020123  040515  020120
 11917  061224  000072
 11918  061226  044510  051040  043505  DH104:  .ASCIZ  /HI REGO LO REGO/
 11919  061234  020060  047514  051040
 11920  061242  043505  000060
 11921  061246  052503  027122  052440  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
 11922  061254  044516  052502  020123
 11923  061262  040515  020120  000072
 11924  061270  045522  051503  020061  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
 11925  061276  020040  045522  051503
 11926  061304  020062  020040  045522
 11927  061312  041504  020040  020040
 11928  061320  045522  040504  020040
 11929  061326  020040  045522  041527
 11930  061334  020122  020040  045522
 11931  061342  040502  020040  020040
 11932  061350  045522  051501  043117
 11933  061356     000
 11934  061357     122  042113  020123  DH202:  .ASCIZ  /RKDS    RKER/
 11935  061364  020040  051040  042513
 11936  061372  000122
 11937  061374  030101  020060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
 11938  061402  020040  030102  020060
 11939  061410  020040  020040  030101
 11940  061416  020061  020040  020040
 11941  061424  030102  020061  020040
 11942  061432  020040  030101  020062
 11943  061440  020040  020040  030102
```

```
11944  061446  020062  020040  020040
11945  061454  030101  020063  020040
11946  061462  020040  030102  000063
11947  061470  042510  042101  051105  DH601:  .ASCIZ  /HEADER SHOULD BE:/
11948  061476  051440  047510  046125
11949  061504  020104  042502  000072
11950  061512  042510  042101  051105  DH602:  .ASCIZ  /HEADER OF SECTOR 0 THIS CYLINDER IS:/
11951  061520  047440  020106  042523
11952  061526  052103  051117  030040
11953  061534  052040  044510  020123
11954  061542  054503  044514  042116
11955  061550  051105  044440  035123
11956  061556     000
11957  061557     120  041501  020113  DH604:  .ASCIZ  /PACK ADDRESS OF ERROR(S) :/
11958  061564  042101  051104  051505
11959  061572  020123  043117  042440
11960  061600  051122  051117  051450
11961  061606  020051  000072
11962  061612  054503  047114  051104  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
11963  061620  020040  051124  041501
11964  061626  020113  020040  042523
11965  061634  052103  051117     000
11966  061641     103  046131  042116  DH6042: .ASCIZ  /CYLNDR  TRACK/
11967  061646  020122  052040  040522
11968  061654  045503     000
11969  061657     105  041503  042040  DH605:  .ASCIZ  /ECC DATA IS:/
11970  061664  052101  020101  051511
11971  061672  000072
11972  061674  047520  020123  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
11973  061702  020040  040520  020124
11974  061710  020040  020040  047503
11975  061716  051122  041505  040524
11976  061724  046102  037505     000
11977  061731     127  051117  030504  DH606:  .ASCIZ  /WORD1   WORD2   WORD3/
11978  061736  020040  053440  051117
11979  061744  031104  020040  053440
11980  061752  051117  031504     000
11981  061757     102  042101  044040  DH607:  .ASCIZ  /BAD HEADER IS :/
11982  061764  040505  042504  020122
11983  061772  051511  035040     000
11984  061777     127  020104  020043  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
11985  062004  020040  043440  047517
11986  062012  020104  020040  041040
11987  062020  042101  020040  020040
11988  062026  044040  020111  044120
11989  062034  020131  046040  020117
11990  062042  044120  020131  053040
11991  062050  052122  040440  020104
11992  062056  045440  050111  051101
11993  062064  000066
11994  062066  047507  042117  020040  DH702:  .ASCIZ  /GOOD    BAD/
11995  062074  020040  040502  000104
11996  062102  040506  046111  047111  DH703:  .ASCIZ  /FAILING DATA WORD :/
11997  062110  020107  040504  040524
11998  062116  053440  051117  020104
11999  062124  000072
```

```
 12000  062126  051123  004460  051123  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
 12001  062134  004461  051123  004462
 12002  062142  051123  000063
 12003  062146  052113  030461  051040  DH705:  .ASCIZ  /KT11 REGS :/
 12004  062154  043505  020123  000072
 12005  062162  042515  020115  042522  DH706:  .ASCIZ  /MEM REGS :/
 12006  062170  051507  035040     000
 12007  062175     120  004503  047514  DH707:  .ASCIZ  /PC     LOERAD  HIERAD  MEMSYS/
 12008  062202  051105  042101  044011
 12009  062210  042511  040522  004504
 12010  062216  042515  051515  051531
 12011  062224     000
 12012  062225     120  004503  051503  DH710:  .ASCIZ  /PC     CSR AD  CSR/
 12013  062232  020122  042101  041411
 12014  062240  051123     000
 12015  062243     103  046131  047111  DH711:  .ASCIZ  /CYLINDERS :/
 12016  062250  042504  051522  035040
 12017  062256     000
 12018  062257     122  042113  020123  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
 12019  062264  020040  051040  042513
 12020  062272  020122  020040  051040
 12021  062300  046513  031122  020040
 12022  062306  051040  046513  031522
 12023  062314     000
 12024  062315     105  051122  051117  DH502:  .ASCIZ  /ERROR TRYING TO GET FAILURE INFO/
 12025  062322  052040  054522  047111
 12026  062330  020107  047524  043440
 12027  062336  052105  043040  044501
 12028  062344  052514  042522  044440
 12029  062352  043116  000117
 12030  062356  042523  047503  042116  DH503:  .ASCIZ  /SECOND TIMEOUT OCCURRED GETTING DRIVE STATUS/
 12031  062364  052040  046511  047505
 12032  062372  052125  047440  041503
 12033  062400  051125  042522  020104
 12034  062406  042507  052124  047111
 12035  062414  020107  051104  053111
 12036  062422  020105  052123  052101
 12037  062430  051525     000
 12038  062433     116  046525  042502  DH800:  .ASCIZ  /NUMBER OF RETRIES:/
 12039  062440  020122  043117  051040
 12040  062446  052105  044522  051505
 12041  062454  000072
 12042                                  .EVEN
 12043  062456  001116  005500  005476  DT100:  .WORD   $ERRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
 12044  062464  001162  001164  001166
 12045  062472  001170
 12046  062474  001256  001260          DT102:  .WORD   $REG36,$REG37
 12047  062500  001174  001176  001200  DT201:  .WORD   $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
 12048  062506  001202  001204  001206
 12049  062514  001210
 12050  062516  001212  001214          DT202:  .WORD   $REG14,$REG15
 12051  062522  001216  001220  001222  DT203:  .WORD   $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
 12052  062530  001224  001226  001230
 12053  062536  001232  001234
 12054  062542  001174  001176  001200  DT601:  .WORD   $REG5,$REG6,$REG7
 12055  062550  001202  001204  001206  DT602:  .WORD   $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
```

```
 12056  062556  001210  001212  001214
 12057  062564  001216
 12058  062566  005254  005252          DT103:  .WORD   PRMPHO,PRMPLC
 12059
 12060  062572  000006                  DF01:   .WORD   6               ;NUMBER OF HEADER LINES
 12061  062574     000                          .BYTE   0               ;NUMBER OF COL FOR FIRST HDR
 12062  062575     000                          .BYTE   0               ;ALL CHARACTERS OCTAL
 12063  062576  061076                          .WORD   DH101           ;SECOND HDR LINE
 12064  062600     007     000                  .BYTE   7,0             ;NUM OF COL-ALL OCTAL
 12065  062602  061165                          .WORD   DH102
 12066  062604     002     000                  .BYTE   2,0
 12067  062606  060737                          .WORD   DH200
 12068  062610     000     000                  .BYTE   0,0
 12069  062612  061270                          .WORD   DH201
 12070  062614     007     000                  .BYTE   7,0
 12071  062616  060761                          .WORD   DH500
 12072  062620     000     000                  .BYTE   0,0
 12073
 12074  062622  000007                  DF02:   .WORD   7
 12075  062624     000     000                  .BYTE   0,0
 12076  062626  061076                          .WORD   DH101
 12077  062630     007     000                  .BYTE   7,0
 12078  062632  061165                          .WORD   DH102
 12079  062634     002     000                  .BYTE   2,0
 12080  062636  060737                          .WORD   DH200
 12081  062640     000     000                  .BYTE   0,0
 12082  062642  061270                          .WORD   DH201
 12083  062644     007     000                  .BYTE   7,0
 12084  062646  061357                          .WORD   DH202
 12085  062650     002     000                  .BYTE   2,0
 12086  062652  061374                          .WORD   DH203
 12087  062654     010     000                  .BYTE   10,0
 12088
 12089  062656  000010                  DF03:   .WORD   10
 12090  062660     000     000                  .BYTE   0,0
 12091  062662  061076                          .WORD   DH101
 12092  062664     007     000                  .BYTE   7,0
 12093  062666  061165                          .WORD   DH102
 12094  062670     002     000                  .BYTE   2,0
 12095  062672  060737                          .WORD   DH200
 12096  062674     000     000                  .BYTE   0,0
 12097  062676  061270                          .WORD   DH201
 12098  062700     007     000                  .BYTE   7,0
 12099  062702  061017                          .WORD   DH501           ;"THE FOLLOWING REG DATA MB INCORRECT"
 12100  062704     000     000                  .BYTE   0,0
 12101  062706  061357                          .WORD   DH202
 12102  062710     002     000                  .BYTE   2,0
 12103  062712  061374                          .WORD   DH203
 12104  062714     010     000                  .BYTE   10,0
 12105
 12106  062716  000003                  DF04:   .WORD   3
 12107  062720     000     000                  .BYTE   0,0
 12108  062722  061076                          .WORD   DH101
 12109  062724     007     000                  .BYTE   7,0
 12110  062726  061165                          .WORD   DH102
 12111  062730     002     000                  .BYTE   2,0
```

```
12112
12113  062732  000007                DFC5:   .WORD   7               ;OPI, HVRC
12114  062734     000      000               .BYTE   0,0
12115  062736  061076                       .WORD   DH101
12116  062740     007      000               .BYTE   7,0
12117  062742  061165                       .WORD   DH102
12118  062744     002      000               .BYTE   2,0
12119  062746  061470                       .WORD   DH601
12120  062750     000      000               .BYTE   0,0
12121  062752  061731                       .WORD   DH606
12122  062754     003      000               .BYTE   3,0
12123  062756  061512                       .WORD   DH602
12124  062760     000      000               .BYTE   0,0
12125  062762  061731                       .WORD   DH606
12126  062764     003      000               .BYTE   3,0
12127
12128  062766  000007                DF07:   .WORD   7
12129  062770     000      000               .BYTE   0,0
12130  062772  061076                       .WORD   DH101
12131  062774     007      000               .BYTE   7,0
12132  062776  061165                       .WORD   DH102
12133  063000     002      000               .BYTE   2,0
12134  063002  061557                       .WORD   DH604
12135  063004     000      000               .BYTE   0,0
12136  063006  061612                       .WORD   DH6041
12137  063010     003      000               .BYTE   3,0
12138  063012  061657                       .WORD   DH605
12139  063014     000      000               .BYTE   0,0
12140  063016  061674                       .WORD   DH6051
12141  063020     003      000               .BYTE   3,0
12142
12143  063022  000005                DF10:   .WORD   5
12144  063024     000      000               .BYTE   0,0
12145  063026  061076                       .WORD   DH101
12146  063030     007      000               .BYTE   7,0
12147  063032  061165                       .WORD   DH102
12148  063034     002      000               .BYTE   2,0
12149  063036  061557                       .WORD   DH604
12150  063040     000      000               .BYTE   0,0
12151  063042  061612                       .WORD   DH6041
12152  063044     003      000               .BYTE   3,0
12153
12154  063046  000004                DF11:   .WORD   4
12155  063050     000      000               .BYTE   0,0
12156  063052  061557                       .WORD   DH604
12157  063054     000      000               .BYTE   0,0
12158  063056  061612                       .WORD   DH6041
12159  063060     003      000               .BYTE   3,0
12160  063062  061777                       .WORD   DH701
12161  063064     000      000               .BYTE   0,0
12162
12163  063066  000006                DF12:   .WORD   6
12164  063070     000      000               .BYTE   0,0
12165  063072  061076                       .WORD   DH101
12166  063074     007      000               .BYTE   7,0
12167  063076  061165                       .WORD   DH102
```

```
12168  063100  002  000                .BYTE   2,0
12169  063102  060737                  .WORD   DH200
12170  063104  000  000                .BYTE   0,0
12171  063106  061270                  .WORD   DH201
12172  063110  007  000                .BYTE   7,0
12173  063112  062257                  .WORD   DH204
12174  063114  004  000                .BYTE   4,0
12175
12176  063116  000006           DF13:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12177  063120  000  000                .BYTE   0,0           ;IN HEADER COMPARE ERROR
12178  063122  061076                  .WORD   DH101         ;AND 2ND LEVEL HEADER
12179  063124  007  000                .BYTE   7,0           ;VRC ERROR
12180  063126  061165                  .WORD   DH102
12181  063130  002  000                .BYTE   2,0
12182  063132  061470                  .WORD   DH601
12183  063134  000  000                .BYTE   0,0
12184  063136  061731                  .WORD   DH606
12185  063140  003  000                .BYTE   3,0
12186  063142  062315                  .WORD   DH502
12187  063144  000  000                .BYTE   0,0
12188
12189  063146  000006           DF14:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12190  063150  000  000                .BYTE   0,0           ;IN OPERATION INCOMPLETE ERROR
12191  063152  061076                  .WORD   DH101
12192  063154  007  000                .BYTE   7,0
12193  063156  061165                  .WORD   DH102
12194  063160  002  000                .BYTE   2,0
12195  063162  061470                  .WORD   DH601
12196  063164  000  000                .BYTE   0,0
12197  063166  061731                  .WORD   DH606
12198  063170  003  000                .BYTE   3,0
12199  063172  062315                  .WORD   DH502
12200  063174  000  000                .BYTE   0,0
12201
12202  063176  000011           DF15:  .WORD   11
12203  063200  000  000                .BYTE   0,0
12204  063202  061076                  .WORD   DH101
12205  063204  007  000                .BYTE   7,0
12206  063206  061165                  .WORD   DH102
12207  063210  002  000                .BYTE   2,0
12208  063212  060737                  .WORD   DH200
12209  063214  000  000                .BYTE   0,0
12210  063216  061270                  .WORD   DH201
12211  063220  007  000                .BYTE   7,0
12212  063222  061357                  .WORD   DH202
12213  063224  002  000                .BYTE   2,0
12214  063226  062356                  .WORD   DH503
12215  063230  000  000                .BYTE   0,0
12216  063232  061017                  .WORD   DH501
12217  063234  000  000                .BYTE   0,0
12218  063236  061374                  .WORD   DH203
12219  063240  010  000                .BYTE   10.0
12220
12221  063242  000011           DF16:  .WORD   11
12222  063244  000  000                .BYTE   0,0
12223  063246  061076                  .WORD   DH101
```

```
12224  063250  007     000             .BYTE   7,0
12225  063252  061165                  .WORD   DH102
12226  063254  002     000             .BYTE   2,0
12227  063256  060737                  .WORD   DH200
12228  063260  000     000             .BYTE   0,0
12229  063262  061270                  .WORD   DH201
12230  063264  007     000             .BYTE   7,0
12231  063266  061357                  .WORD   DH202
12232  063270  002     000             .BYTE   2,0
12233  063272  062315                  .WORD   DH502
12234  063274  000     000             .BYTE   0,0
12235  063276  061017                  .WORD   DH501
12236  063300  000     000             .BYTE   0,0
12237  063302  061374                  .WORD   DH203
12238  063304  010     000             .BYTE   10,0
12239
12240  063306  000005          DF17:   .WORD   5
12241  063310  000     000             .BYTE   0,0
12242  063312  061076                  .WORD   DH101
12243  063314  007     000             .BYTE   7,0
12244  063316  061165                  .WORD   DH102
12245  063320  002     000             .BYTE   2,0
12246  063322  062243                  .WORD   DH711
12247  063324  000     000             .BYTE   0,0
12248  063326  062066                  .WORD   DH702
12249  063330  002     000             .BYTE   2,0
12250
12251  063332  000002          DF20:   .WORD   2
12252  063334  000     000             .BYTE   0,0
12253  063336  061226                  .WORD   DH104
12254  063340  002     000             .BYTE   2,0
12255
12256  063342  000002          DF21:   .WORD   2
12257  063344  000     000             .BYTE   0,0
12258  063346  061674                  .WORD   DH6051
12259  063350  003     000             .BYTE   3,0
12260
12261  063352  000006          DF22:   .WORD   6
12262  063354  000     000             .BYTE   0,0
12263  063356  060672                  .WORD   DH100
12264  063360  000     000             .BYTE   0,0
12265  063362  061076                  .WORD   DH101
12266  063364  007     000             .BYTE   7,0
12267  063366  061165                  .WORD   DH102
12268  063370  002     000             .BYTE   2,0
12269  063372  061246                  .WORD   DH106
12270  063374  000     000             .BYTE   0,0
12271  063376  061226                  .WORD   DH104
12272  063400  002     000             .BYTE   2,0
12273
12274  063402  000002          DF23:   .WORD   2
12275  063404  000     000             .BYTE   0,0
12276  063406  062433                  .WORD   DH800
12277  063410  001     000             .BYTE   1,0
12278
12279  063412  000001          DF24:   .WORD   1
```

```
12280  063414      002      000                    .BYTE   2,0
12281
12282  063416  000000                      DF25:   .WORD   0
12283  063420      007      000                    .BYTE   7,0
12284
12285  063422  000002                      DF26:   .WORD   2
12286  063424      002      000                    .BYTE   2,0
12287  063426  061374                              .WORD   DH203
12288  063430      010      000                    .BYTE   10,0
12289
12290  063432  000005                      DF27:   .WORD   5
12291  063434      000      000                    .BYTE   0,0
12292  063436  061076                              .WORD   DH101
12293  063440      007      000                    .BYTE   7,0
12294  063442  061165                              .WORD   DH102
12295  063444      002      000                    .BYTE   2,0
12296  063446  062102                              .WORD   DH703
12297  063450      000      000                    .BYTE   0,0
12298  063452  062066                              .WORD   DH702
12299  063454      002      000                    .BYTE   2,0
12300
12301  063456  000005                      DF30:   .WORD   5
12302  063460      000      000                    .BYTE   0,0
12303  063462  061076                              .WORD   DH101
12304  063464      007      000                    .BYTE   7,0
12305  063466  061165                              .WORD   DH102
12306  063470      002      000                    .BYTE   2,0
12307  063472  062146                              .WORD   DH705
12308  063474      000      000                    .BYTE   0,0
12309  063476  062126                              .WORD   DH704
12310  063500      004      000                    .BYTE   4,0
12311
12312  063502  000005                      DF31:   .WORD   5
12313  063504      000      000                    .BYTE   0,0
12314  063506  061076                              .WORD   DH101
12315  063510      007      000                    .BYTE   7,0
12316  063512  061165                              .WORD   DH102
12317  063514      002      000                    .BYTE   2,0
12318  063516  062162                              .WORD   DH706
12319  063520      000      000                    .BYTE   0,0
12320  063522  062225                              .WORD   DH710
12321  063524      003      000                    .BYTE   3,0
12322
12323  063526                      RWBUF:                  ;READ/WRITE DATA BUF <AT LEAST 1536(DEC) WORDS>
12324
12325  063526  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
12326  063534  020040  020040  020040
12327  063542  025052  020052  047516
12328  063550  042524  025040  025052
12329  063556  005015      012
12330  063561      101  046114  042040          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12331  063566  044522  042526  020123
12332  063574  047524  041040  020105
12333  063602  042524  052123  042105
12334  063610  046440  051525  020124
12335  063616  040510  042526  035040
```

```
12336  063624  005015     012
12337  063627     061  020056  020101          .ASCII  /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12338  063634  030062  047440  020122
12339  063642  031062  051440  041505
12340  063650  047524  020122  047506
12341  063656  046522  052101  042524
12342  063664  020104  040503  052122
12343  063672  044522  043504  006505
12344  063700     012
12345  063701     062  020056  042510          .ASCII  /2. HEADS MANUALLY LOADED/<15><12>
12346  063706  042101  020123  040515
12347  063714  052516  046101  054514
12348  063722  046040  040517  042504
12349  063730  006504     012
12350  063733     063  020056  042504          .ASCII  /3. DESIRED PORT SELECTED/<15><12>
12351  063740  044523  042522  020104
12352  063746  047520  052122  051440
12353  063754  046105  041505  042524
12354  063762  006504     012
12355  063765     064  020056  051127          .ASCII  /4. WRITE LOCK DISABLED/<15><12>
12356  063772  052111  020105  047514
12357  064000  045503  042040  051511
12358  064006  041101  042514  006504
12359  064014     012
12360  064015     065  020056  051104          .ASCII  /5. DRIVE READY LIGHT ON/<15><12><12>
12361  064022  053111  020105  042522
12362  064030  042101  020131  044514
12363  064036  044107  020124  047117
12364  064044  005015     012
12365  064047     104  044522  042526          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12366  064054  020123  047516  020124
12367  064062  047524  041040  020105
12368  064070  042524  052123  042105
12369  064076  046440  051525  020124
12370  064104  040510  042526  041040
12371  064112  052117  006510     012
12372  064117     120  051117  051524          .ASCII  /PORTS DE-SELECTED./<15><12><12>
12373  064124  042040  026505  042523
12374  064132  042514  052103  042105
12375  064140  006456  005012
12376  064144  046120  040505  042523          .ASCII  &PLEASE NOTE THAT THE DEFAULT VERSION OF READ/WRITE&<15><12>
12377  064152  047040  052117  020105
12378  064160  044124  052101  052040
12379  064166  042510  042040  043105
12380  064174  052501  052114  053040
12381  064202  051105  044523  047117
12382  064210  047440  020106  042522
12383  064216  042101  053457  044522
12384  064224  042524  005015
12385  064230  040504  040524  052040          .ASCII  /DATA TEST 21 TAKES ABOUT 10 MINUTES TO COMPLETE,/<15><12>
12386  064236  051505  020124  030462
12387  064244  052040  045501  051505
12388  064252  040440  047502  052125
12389  064260  030440  020060  044515
1239C  064266  052516  042524  020123
1239:  064274  047524  041440  046517
```

```
12392  064302  046120  052105  026105
12393  064310  005015
12394  064312  042522  040507  042122          .ASCII   REGARDLESS OF THE PASS NUMBER./<15><12><12>
12395  064320  042514  051523  047440
12396  064326  020106  044124  020105
12397  064334  040520  051523  047040
12398  064342  046525  042502  027122
12399  064350  005015     012
12400  064353     106  051117  032440          .ASCIZ  /FOR 50 HZ OPERATION, PATCH LOC. 166 = 1./<15><12><12>
12401  064360  020060  055110  047440
12402  064366  042520  040522  044524
12403  064374  047117  020054  040520
12404  064402  041524  020110  047514
12405  064410  027103  030440  033066
12406  064416  036440  030440  006456
12407  064424  005012     000
12408
12409
12410
12411          000001                    .END
```

C03


```
ABASE = 177440        2010#   2102    2143    4523
ABORT = 000200        4047#   9131    9136    9229    9254    9342    9352    9357    9362    9367    9375    9391    9391
                      9425
ABOVE   012062        4336#   5749
ABOVE1  012117        4341#   5841
ABOVE3  012165        4348#   5993
ABVMX1  003162        3450#   5722*   5747    5811*   5964*   7653*   7834
ABVMX2  003164        3451#   5831*   5982*   7654*   7849
ACDW1 = 000000        2102
ACDW2 = 000000        2102
ACLO  = 000010        3051#   9191
ACPUOP= 000000        2102    2117
ADDW0 = 000000        2102
ADDW1 = 000000        2102
ADDW10= 000000        2102
ADDW11= 000000        2102
ADDW12= 000000        2102
ADDW13= 000000        2102
ADDW14= 000000        2102
ADDW15= 000000        2102
ADDW2 = 000000        2102
ADDW3 = 000000        2102
ADDW4 = 000000        2102
ADDW5 = 000000        2102
ADDW6 = 000000        2102
ADDW7 = 000000        2102
ADDW8 = 000000        2102
ADDW9 = 000000        2102
ADEVCT= 000000        2102    2108
ADEVM = 000377        2011#   2102    2144
AENV  = 000000        2102    2113
AENVM = 000000        2102    2114
AFATAL= 000000        2102    2105
ALDRVS  010076        4141#   4591
ALLDRV  014376        4529    4600    4701#
ALLTRM  043224        9230    9365    9372#
ALNCYL= 000365        4040#
ALNPAK  007623        4109#   6958
AMADR1= 000000        2102    2130
AMADR2= 000000        2102    2134
AMADR3= 000000        2102    2137
AMADR4= 000000        2102    2140
AMAMS1= 000000        2102    2124
AMAMS2= 000000        2102    2132
AMAMS3= 000000        2102    2135
AMAMS4= 000000        2102    2138
AMSGAD= 000000        2102    2110
AMSGLG= 000000        2102    2111
AMSGTY= 000000        2102    2104
AMTYP1= 000000        2102    2125
AMTYP2= 000000        2102    2133
AMTYP3= 000000        2102    2136
AMTYP4= 000000        2102    2139
ANYDER= 100000        4053#   6950    8846    9413
APASS = 000000        2102    2107
APRIOR= 000000        2102
```

```
APTCSU= 000040          10747  11407#
APTENV= 000001          10740  11186  11363  11405#
APTSIZ= 000200           4451  11404#
APTSPC= 000100          10742  11365  11406#
ASKMDE  015304           4871#  4877   4878   4883   4898   4903   4908   4925   4947   4971   4985   4996   5005
                         5021   5040   5049   5075   5098   5170   7283
ASKTMO  014540           4753#  4756   4757   4762   4786   4794   4800   4816   4827   4836   4861
ASKTST  014534           4667   4752#
ASWREG= 000000           2102   2115
AS2SP2  055572           8893   11575#
ATESTN= 000030           2102   2106
AUNIT = 000000           2102   2109
AUSWR = 000000           2102   2116
AVECT1= 120210           2009#  2102   2141   4522
AVECT2= 000000           2102   2142
AVREG   012012           4328#  7919
AVRSEK  011601           4301#  5869
A.ABNL  003036           3332#  6846*  6909*  6913*  8369*  8458*  8475*  8484*  9107*  9373*  9386*  9388*  10309
A.CONT  003040           3333#  10318
A.NORM  003034           3331#  6847*  9108*  9374*  10312
BADDRV  007540           4099#  6916*  6917*  6919   6927*  6928*  6929   6935*  6936*  6937   6955*  6956*  6957
BADSEC= 001000           4049#  8177   8747   8755   9319   9418
BAD632  012316           4365#  6952
BAI   = 000020           3010#  10477
BA16  = 000400           2986#
BA17  = 001000           2987#
BDSCHD  036504           8458   8475   8489#
BDSECT  011423           4279#  8508
BDSRCK  037556           8175   8737#  9318
BELOW   012025           4331#  5743
BGNRTY  043400           9031   9384   9390   9411#
BIT0  = 000001           1901#  2983   3000   3026   3048   3196   3340   3433   4608   4706   6061   6576   7322
                         8051   8068   8083   8122   8139   8193   8235   8990   10592
BIT00 = 000001           1891#  1901
BIT01 = 000002           1890#  1900   5621
BIT02 = 000004           1889#  1899
BIT03 = 000010           1888#  1898   5736   5838   5990
BIT04 = 000020           1887#  1897
BIT05 = 000040           1886#  1896
BIT06 = 000100           1885#  1895   7637
BIT07 = 000200           1884#  1894   5366   7450
BIT08 = 000400           1883#  1893   7478   7510
BIT09 = 001000           1882#  1892   5315   5607   11196  11327
BIT:  = 000002           1900#  3001   3028   3197   3341   3434   4041   6229   8994
BIT10 = 002000           1881#  2988   3017   3039   3074   3088   3101   3116   3130   3209   3350   4050   9023
                         11174
BIT11 = 004000           1880#  2989   3018   3040   3059   3075   3089   3102   3117   3131   3210   4051   11334
BIT12 = 010000           1879#  2991   3019   3041   3076   3090   3103   3104   3118   3132   3211   4052   8905
                         8933
BIT13 = 020000           1878#  2992   3020   3042   3060   3077   3091   3105   3119   3133   3212   6826   6828
                         8074   8077   8148   8151   11181
BIT14 = 040000           1877#  2993   3021   3043   3061   3078   3092   3106   3120   3134   3213   3351   11309
BIT15 = 100000           1876#  2994   2995   3022   3044   3062   3079   3139   3214   3352   4053   4982   7213
                         9026
BIT2  = 000004           1899#  3002   3029   3030   3050   3198   3342   3435   4042   6235   8998
BIT3  = 000010           1898#  3008   3009   3031   3051   3199   3343   3436   4043   6231   9005
```

E03


```
BIT4   = 000020   1897* 3003  3010  3032  3052  3053  3068  3110  3124  3200  3344  3437  4044
                  5233  6223  9009
BIT5   = 000040   1896* 3011  3012  3033  3054  3065  3083  3096  3111  3125  3201  3345  4045
                  6172  6219  6365  8870  9013  10389
BIT6   = 000100   1895* 2984  3013  3034  3055  3070  3084  3097  3112  3126  3202  3347  4046
                  4793  4984  5004  6069  6301  6370  6403  8220
BIT7   = 000200   1894* 2985  3014  3035  3056  3057  3071  3085  3098  3113  3127  3204  4047
                  4520  4703  4715
BIT8   = 000400   1893* 2986  3015  3036  3037  3058  3072  3086  3099  3114  3128  3205  3348
                  4048  9017
BIT9   = 001000   1892* 2987  3016  3038  3073  3087  3100  3115  3129  3206  3349  4049  5218
                  6542  9020  11286
BKSLSH  012417    4377* 7079  7087
B_DEXM  044230    9270  9295  9559*
B_WMIN  003160    3449* 5716* 5741  7652*
BPTVEC= 000014    1908*
BSE    = 000200   3035* 8489  9310  9742  9908  10144
BSERR  = 000002   4041* 6051  8669  9312  9322  9411
BSFACT  004222    3467* 8457  8506  8738
BSSOFT  003222    3466* 8469  8538  8743  8745
BUFFO   005262    3474* 4566  4598  4601  4650  4671  4675  4677  4682  4693  4697  4760  4763
                  4766  4769  4804  4806  4837  4840  4842* 4847  4865  4881  4884  4887  4890
                  4893  4939  4942  4945  5025  5027  5059  5076  5102  5109  5119  5121  5124
                  5129  5138  5176  5179  6487  6491  7082  7096* 7101* 7102  7106* 7223  7225
                  7245  7248  7250* 7255  7271
BUFPRT  005472    3427*
B.BA16= 000001    3000* 10471
B.BA17= 000002    3001* 10471
B.CDT  = 000004   3002* 10125 10471
B.CFMT= 000020    3003*  5208  5210  5212  5220  5652  5660  5682  9509  9511  9513  9517  10125
                 10471  10555
CALBRT  032260    5645  5787  5868  5939  7564*
CART    011354    4271* 7427
CCLR   = 100000   2995* 10078
CDT    = 002000   2988* 9884  10434 10491 10542 10565
CERR   = 100000   2994* 9279  9304  9780  9839  9898  10080 10131 10446 10502 10590
CFMT   = 010000   2991* 5661  5683  9538  9571  9884  10434 10491 10542 10565
CHGTST  014756    4768  4797* 4801  4808
CHKITR  030100    5253  5277  5303  5351  5390  5438  5494  5530  5554  5587  5637  5779  5860
                  5931  6008  6091  6166  6995*
CHKLIM  035330    6184  6211  8253*
CHKLST  014470    4573  4656  4699  4716  4719*
CHKPRM  031200    5065  5144  7304*
CHKXOP  026556    4927  6701*
CLEAR  = 000105   2954*
CLKFAL  011531    4294* 7576
CLKOF   032236    7541* 7618
CLKON   032162    7520* 7571
CLOCK   032232    4499  7535*
CLR    = 000040   3011*
CLRPRM  027372    6851  6868*
CMDTO  = 000100   3202* 9154  9475  9641
CMPBUF  034422    6237  8109*
CMPTIM  033172    5804  5824  5957  5975  7772*
CMSTRT  012542    4402  4406  4411* 9402
CNTRCV  007774    4129* 4652  8382
```

```
CNTRLC  012362          4371#   6369    7050
CNTRLG  012406          4375#   6383    7068
CNTRLR  012374          4373#   6379
CNTRLU  012401          4374#   7062
CNTRLZ  012367          4372#   6375    7056
CNVTIM  033504          5689    7695    7872#
COE   = 001000          3039#   9236    9742    9908    10144
COMMA   012421          4378#   4613
COMSTR  005236          3469#   9811    8826    9430
CONCLR= 000176          2972#  10573
CONERR  040650          3333    8986#
CR    = 000015          1816#  10786   10796
CRLF  = 000200          1817#  10757   10796
CRMPHO  005260          3473#   8183    8821    8823*
CRMPLO  005256          3472#   8184    8822    8824*
CRTSER  031624          5232    7416#
CR2LF   012413          4376#   8892    8904    8928
CS      005750          3586#   5233    5736    5838    5990    8870
CTLOUT  035700          5694    5801    5821    5954    5972    8353#   8782
CTO   = 004000          2989#   9199   10140
CYLNDR  005504          3482#   5363*   5366    5369    5371*   5536    5592*   5599    5606*   5607    5610*   5617    5621
                        7506*   7507*   7508    7510*
C.INIT  050006          5664    7691    8787    8797   10351#
C.RTRN  051660         10455   10483   10511   10518   10521   10528   10532   10549   10571   10576   10586   10595   10608
                       10612   10617#
C.SPEC  050700         10391   10487#
DASTRT  012512          1973    4404#
DCK   = 100000          3044#   8489    9324    9742    9908    10144
DCKERR= 000020          4044#   9326    9411
DCLO  = 000020          3053#
DDISP = 177570          1823#   2040    4439
DDT   = 000400          3058#
DERCNT  003111          3409#   9263    9288    9333    9344
DESL  = 000010          3008#
DFLTST  005650          3546#   3566    4791    4810    4813    4834    4859
DFQUES  010756          4225#   5018
DFSTRT  012456          1957    4396#
DFTEST  010311          4168#   4797
DF01    062572          2167    2173    2179    2185    2227    2335    2605    2641   12060#
DF02    062622          2191    2209    2215    2221    2233    2239    2245    2251    2257    2263    2269    2275    2281
                        2287    2293    2299    2353    2359    2365    2473    2479    2485    2491    2497    2503    2509
                        2515    2521    2527    2569    2587    2611   12074#
DF03    062656          2197    2203    2575    2581   12089#
DF04    062716          2329   12106#
DF05    062732          2305    2311   12113#
DF07    062766          2317   12128#
DF10    063022          2323   12143#
DF11    063046          2623   12154#
DF12    063066          2371    2377    2383    2389    2395    2401    2407    2413   12163#
DF13    063116          2419    2431   12176#
DF14    063146          2425   12189#
DF15    063176          2437   12202#
DF16    063242          2443   12221#
DF17    063306          2461   12240#
DF20    063332          2629   12251#
DF21    063342          2539   12256#
```

```
DF22    063352          2635   12261#
DF23    063402          2545    2551    2557   12274#
DF24    063412          2563   12279#
DF25    063416          2467    6058#   8111#   8165#  12282#
DF26    063422          2617   12285#
DF27    063432          2593   12290#
DF30    063456          2647    6560#  12301#
DF31    063502          2653    6603#   6604#  12312#
DH100   060672          2165    2171    2177    2183    2189    2195    2201    2207    2213    2219    2225    2231    2237
                        2243    2249    2255    2261    2267    2273    2279    2285    2291    2297    2303    2309    2315
                        2321    2327    2333    2351    2357    2363    2369    2375    2381    2387    2393    2399    2405
                        2411    2417    2423    2429    2435    2441    2459    2471    2477    2483    2489    2495    2501
                        2507    2513    2519    2525    2567    2573    2579    2585    2591    2603    2609    2639    2645
                        2651   11876#  12263
DH101   061076          8909   11901#  12063   12076   12091   12108   12115   12130   12145   12165   12178   12191   12204
                       12223   12242   12265   12292   12303   12314
DH102   061165          8920   11911#  12065   12078   12093   12110   12117   12132   12147   12167   12180   12193   12206
                       12225   12244   12267   12294   12305   12316
DH103   061203          2627   11914#
DH104   061226         11918#  12253   12271
DH105   060714          8907   11879#
DH106   061246         11921#  12269
DH200   060737         11883#  12067   12080   12095   12169   12208   12227
DH201   061270         11924#  12069   12082   12097   12171   12210   12229
DH202   061357         11934#  12084   12101   12212   12231
DH203   061374         11937#  12086   12103   12218   12237   12287
DH204   062257         12018#  12173
DH500   060761         11887#  12071
DH501   061017         11893#  12099   12216   12235
DH502   062315         12024#  12186   12199   12233
DH503   062356         12030#  12214
DH601   061470         11947#  12119   12182   12195
DH602   061512         11950#  12123
DH604   061557         11957#  12134   12149   12156
DH6041  061612          8509   11962#  12136   12151   12158
DH6042  061641          2561   11966#
DH605   061657          2537   11969#  12138
DH6051  061674         11972#  12140   12258
DH606   061731         11977#  12121   12125   12184   12197
DH607   061757         11981#
DH701   061777          6057#   8110#   8164#  11984#  12160
DH702   062066          2615   11994#  12248   12298
DH703   062102         11996#  12296
DH704   062126          6561#  12000#  12309
DH705   062146         12003#  12307
DH706   062162         12005#  12318
DH707   062175          6603   12007#
DH710   062225         12012#  12320
DH711   062243         12015#  12246
DH800   062433         12038#  12276
DI    = 040000          2993#   7720    9928   10600
DISPLA  001142          2040#   4439#   4447#  11173#  11347#
DISPRE  000174          1964#   4447
DLT   = 100000          3022#   9256    9736    9866    9902   10142
DLTFLG  003132          3426#
DMC   = 000040          3069#
```

```
DONE    003113          3411#   9783*   8790    8794*   8800    8845*   8987*   9101*   9179*   9444*   9447*
DOTIM   003123          3419#   4494*   4501*   5642    5784    5865    5936    7578*
DRA     = 000001        3048#   9731    10514
DRDY    = 000200        3057#
DRIV    011345          4269#   7385
DRIVE   005500          3480#   4722*   5187*   5194*   5195    5198    5226    6343    6854    6855    6899    6916    6927
                        6935    6955    9176    12043
DRNAFG  003127          3423#   8784*   8792    8795*   9000    9173*
DRNAVL= 010000          4052#
DROPDR  011255          4259#   8876
DROT    = 000040        3054#   9246
DRPAR   = 000010        3031#   9186
DRPDRV= 002000          3209#   10376
DRVCAL  037662          5207    5213    5260    5654    5793    5874    5945    6031    6043    6046    6106    6122    6241
                        6853    6911    6942    6949    6962    6979    7384    7426    7453    7462    7704    7957    8372
                        8411    8414    8460    8477    8665    8780#   9387    9429    9439    9474    9516
DRVDSC= 000040          3201#   9354    9816    9982    10044
DRVERS  003120          3416#   4462*   5192*   8869*   8873    8875*
DRVHRD= 000020          3200#   9349    9823    10018   10149
DRVLST  005576          3510#   4609    4628    4640    4687    4695*   4707*   4710*   4720    4731*   5199
DRVMSK= 000007          3007#   9725    9766    9917    10362
DRVNO   011341          4268#   5226*   5227*
DRVPDT= 000004          3198#   9924    9992
DRVPOS= 000002          3197#   9924    9951    10413
DRVSEQ  007522          4096#   4605
DRVSER  031504          5231    7381#
DRVSZD= 010000          3211#   9167    9754    10516
DRVTST  013660          4399    4571    4587#   4593    4594    4595    4603    4660    4661    4755    4783    4799    4826
                        4876    4968    4993    5020    5048    5097    5169    7280    8361
DRVUSE= 000001          3196#   10376
DRY     = 000200        3056#
DR.CLR= 000005          2979#   9804    9967    10009   10033   10147
DR.SEL= 000001          2978#   9874    9953    10268   10275   10282   10289   10493
DROXOP  010045          4136#   6906
DSC     = 040000        3061#
DSWR    = 177570        1822#   2039    4438
DTBAII= 100000          3214#   10376   10475
DTE     = 010000        3041#   8489    9251    10144
DTYE    = 000040        3033#   9214    10144
DT10C   062456          2166    2172    2178    2184    2190    2196    2202    2208    2214    2220    2226    2232    2238
                        2244    2250    2256    2262    2268    2274    2280    2286    2292    2298    2304    2310    2316
                        2322    2328    2334    2352    2358    2364    2370    2376    2382    2388    2394    2400    2406
                        2412    2418    2424    2430    2436    2442    2460    2472    2478    2484    2490    2496    2502
                        2508    2514    2520    2526    2568    2574    2580    2586    2592    2604    2610    2634    264C
                        2646    2652    12043#
DT102   062474          12046#
DT103   062566          2628    12058#
DT201   062500          12047#
DT202   062516          2616    12050#
DT203   062522          12051#
DT601   062542          2538    2544    2550    2556    2562    2622    12054#
DT602   062550          2466    12055#
DUACES  010571          4201#   4473
DULACS  003126          3422#   4397*   4404*   4409*   4471    4665    5237    6238    6703    6997    9169
DUMPAS  024764          5197    6300#
DZR6M   007176          4055#   4464
```

```
ECCNC = 020040          4045*   9329
ECCW  = 020000          3077*
ECH   = 000100          3034*   9327    9742    9908    10144
ECOBAD  025350          4787    4972    4997    6417*
EMTVEC= 000030          1911*   4422*   4423*
EM1     055577          2164   11576*
EM10    056012          2206   11603*
EM100   060123          2554   11810*
EM101   060146          2560   11814*
EM102   060224          2566   11822*
EM103   060276          2572   11829*
EM104   060324          2578   11833*
EM105   060346          2584   11836*
EM106   060401          2590   11841*
EM107   060434          2596   11846*
EM11    056021          2212   11605*
EM110   060501          2602   11853*
EM111   060534          2608   11858*
EM112   060555          2614   11861*
EM113   060610          2638   11866*
EM114   060636          2644   11870*
EM115   060653          2650   11873*
EM12    056034          2218   11607*
EM13    056055          2224   11610*
EM14    056077          2230   11614*
EM15    056125          2236   11618*
EM16    056146          2242   11621*
EM17    056163          2248   11624*
EM2     055623          2170   11580*
EM20    056204          2254   11627*
EM21    056221          2260   11630*
EM22    056241          2266   11633*
EM23    056263          2272   11637*
EM24    056314          2278   11642*
EM25    056334          2284   11645*
EM26    056357          2290   11649*
EM27    056371          2296   11651*
EM3     055647          2176   11584*
EM30    056414          2302    2422   11655*
EM31    056441          2308    2428   11659*
EM32    056462          2314   11662*
EM33    056503          2320   11665*
EM34    056525          2326    2626   11669*
EM35    056545          2332   11672*
EM36    056601          2338   11677*
EM37    056634          2344   11682*
EM4     055672          2182   11588*
EM40    056677          2350   11688*
EM41    056743          2356   11695*
EM42    057001          2362   11701*
EM43    057031          2368   11706*
EM5     055713          2188   11591*
EM52    057066          2410   11711*
EM53    057114          2416   11715*
EM56    057141          2434    2440   11719*
EM6     055732          2194   11594*
```

```
EM60    057160          2446  11722#
EM61    057214          2452  11727#
EM62    057262          2458  11734#
EM63    057306          2374   2470  11738#
EM64    057353          2380   2476  11745#
EM65    057416          2386   2482  11751#
EM66    057444          2392   2488  11755#
EM67    057477          2398   2494  11760#
EM7     055756          2200  11598#
EM70    057536          2404   2500  11766#
EM71    057605          2506  11773#
EM72    057645          2512  11779#
EM73    057714          2518  11786#
EM74    057750          2524  11791#
EM75    057770          2530  11794#
EM76    060020          2536  11798#
EM77    060102          2542   2548  11807#
ENBCSR  026056          4460   4590   4872   6570#  6616
ENTLCI  010263          4164#  4753
ENTPAS  011207          4252#  5167
EQUALS  012444          4387#  7182
ERRCNT  003116          3414#  4398#  4570#  6976#  8848#  8850   8853   8855#  8988#  9102#  9400#  9420   9423
ERRCOM  005476          3479#  9039#  9040#  12043
ERRFRE  040106          3331   6847   8845#  9374
ERRHDL  041354          3332   6338   6846   6913   8369   8484   8491   9100#  9373   9388
ERRLMT  003117          3415#  9420
ERRVEC= 000004          1904#  4436   4437#  4448#  4456#  4457#  6555#  6556#  6571   6572   6573#  6582#  6583#
                        6593   6594   6597#  6598#  6613#  6614#  6620#  11314  11315#  11317#  11320#  11428  11429
                       11433#  11438#  11450#  11455#  11464#  11474#  11475#
ER2ENT  041374          9104#  9166   9284   9309   9393
EXPLAN  010415          4181#  4870
E.CCLR= 000001          3340#  10083
E.CERR= 001000          3349#  10157
E.CLAT= 000020          3344#  9809   9973   10014  10038
E.CMTO= 040000          3351#
E.CONT  003042          3334#  8990   8994   8998   9005   9009   9013   9017   9020   9023   9026   9030#  9708#
                        9746#  9809#  9819#  9890#  9912#  9935#  9941#  9973#  10014#  10038#  10083#  10135#  10157#
                       10296#  10530#  10584#  10592#  10605#
E.DLT = 000400          3348#  9890
E.DPAR= 002000          3350#  9819   10296
E.ILLD= 000100          3347#  10584
E.MDS = 100000          3352#  9708   10135  10530
E.NOAT= 000002          3341#  9935
E.SCLR= 000040          3345#  10605
E.UATT= 000004          3342#  9941
E.UDAT= 000010          3343#  9746   9912
E.UNLD= 020000          3212#  9827   9947   10022  10032
FACTBS  011376          4275#  8507
FC      005712          3571#  5283   5396   5399   5409   5412   5416   5444   5447   5449   5459   5462   5466
                        5470   5499   5505   6014   6016   6097   6200   6268
FDTERM  033062          5882   7730#
FINADR  034016          7994#  8170
FINCYL  005564          3501#  8024#  8172   8197   8203
FINMEM  036702          8550#  8600
FINSEC  005567          3503#  8012#  8174   8201   8205
FINTRK  005566          3502#  8022#  8173   8199   8204
```

```
FMTE  = 000020          3032*   9218   10144
FNDXDP  026504          6677*   6725
FORMAT  003115          3413*   5214*   5220*   6024    6856    7416    8004    8263    9452    8472    8577    9509    9518
FORWRD  011714          4316*   5909    7828
FS      005506          3483*   5215*   5221*   6099    6208
FT      005720          3574*   4904    6015    6017    6207
GETAVG  033244          5836    5988    7794*
GETPRM  025532          4539    4544    4553    6473*   6513
GETXDP  027040          4514    6260    6776*   8364
GNS   = ******          1963   11561   11562   11563   11564   11565   11568   11569   11570   11571
GO    = 000001          2983*
GTPKAD  044104          9335    9346    9528*
GTPRMS  037254          8645*   8677
GTSWRG  025642          4489    6509*   7069    8387
GTO     037266          8641    8644    8648*
HCRC  = 000400          3036*
HDR.AD  003060          3374*   9860    9864*  10554*
HDR.CT  003062          3375*   9868*  10557*  10560*
HI.RAD= 177742          1955*   6601
HIGOCT  005470          3476*   5063*   5143*   7153*   7156*   7309    7324    7329    7336
HLPOVL  003136          3431*   4463*
HLTPRG  043364          4633    6563    6612    8383    8468    9399*   9426
HLTRQD  010024          4133*   8381
HT    = 000011          1814*  10755   10796
HVRC  = 000400          3037*   8489    9285    9742    9908   10144
HVRCER= 000004          4042*   9287    9411
HZ      000166          1969*   7525    7607
H.HEAD  003065          3378*
IC      005716          3573*   5401    5405    5411    5415    5451    5455    5461    5465    5507    5508    5512    5513
                        6270    6274
IDAE  = 002000          3039*   9241
IE    = 000100          2984*   4653    6921    7707    9275    9300   10088   10152   10440   10481   10547   10614
ILC   = 000001          3026*   9202   10144
ILF   = 000004          3029*   9210
INCRMA  036174          8423*
INCRSC  036766          8562    8576*   8604
INITMM  025674          6529*   6642    6803
INITSS  027300          4655    5205    5224    5256    5280    5306    5354    5393    5441    5497    5533    5557    5590
                        5640    5782    5863    5918    5934    6011    6094    6169    6846*   6898    7382    7416    8446
INIVRB  032552          5648    5790    5871    5942    7652*
INPUTP  015300          4668    4771    4870*   8377
INTCHR  005522          3489*   4779    4781    4784    4964    4966    4969    4989    4991    4994    5193*   5236*   6366*
                        6385    6402*   6417    8355    8359    8366*   8375    8379    8385
INTMSK  003071          3383*   9177*   9178    9783    9807    9828    9853    9939    9971   10012   10023   10036   10307*
                       10311*  10315*  10364*  10450
INTR  = 000300          2974*
IOTVEC= 000020          1909*   4420*   4421*
IR    = 000100          3013*
IS      005736          3581*   6259
IT      005724          3576*   6263
I.AERR  046370          9959    9965    9983    9992*
I.ATTN  045720          9719    9763    9896*
I.CCLR  046754          9797    9847   10078*  10150   10510   10575
I.CSTS  047406          9790   10225*
I.CSTI  047430          9794    9846   10229*  10499
I.DRV   003072          3387*   9177   10364   10365
```

```
I.ERR   045142            9791    9795#
I.ERRA  045122            9790#   9858
I.ERRC  045130            9728    9781    9793#   9926
I.HDAL  045366            9774    9837#
I.INTR  044506            4477    5754    5763    7700    7719    9698#
I.ISRL  003064            3377#   9712    9715#   9718#   10089#  10386#  10540#  10613#
I.ISSU  047036            9805    9875    9954    9968    10010   10034   10120#  10269   10276   10283   10290
I.IOD   044766            9716    9733    9760#
I.RTRN  046736            9710    9748    9758    9786    9798    9803    9806    9812    9821    9830    9833    9848    9850
                          9876    9879    9888    9892    9914    9930    9937    9943    9955    9969    9976    9988    10008
                         10011   10016   10025   10028   10035   10040   10049#
I.STAT  047512            9802    10007   10266#  10520
I.STOR  047260           10130    10180#
I.UNLD  046636            9949    10032#
KBDHDL  025170            4474    6358#
KIPAR0= 172340            1942#   6531    11414   11442
KIPAR1= 172342            1943#
KIPAR2= 172344            1944#
KIPAR3= 172346            1945#
KIPAR4= 172350            1946#
KIPAR5= 172352            1947#
KIPAR6= 172354            1948#   6811#   6829#   8067#   8076#   8138#   8146    8150#
KIPAR7= 172356            1949#   11458
KIPDR0= 172300            1931#
KIPDR1= 172302            1932#
KIPDR2= 172304            1933#
KIPDR3= 172306            1934#
KIPDR4= 172310            1935#
KIPDR5= 172312            1936#
KIPDR6= 172314            1937#
KIPDR7= 172316            1938#
KTERHD  025762            4479    6551#
LASTWC  005570            3504#   7995    8169#
LC      005714            3572#   5262    5286    5399    5402    5406    5409    5449    5452    5456    5459    5500    5505
                          5509    5514    6268    6271    6275
LCVEC   003150            3444#   4506
LEV2ER= 000400            4048#   9104    9159    9266    9272    9291    9297    9389    9445
LF    = 000012            1815#   10790   10796
LIM1    012270            4360#   5744
LIM2    012303            4362#   5750
LKS     003140            3440#   4505    7522#   7543#
LODBUF  034174            6217    8042#
LODFLS  015022            4637    4805    4810#
LODFPT  016210            4817    5026    5031#
LODP14  033760            6174    7971#
LODSEC  033652            6030    6041    6105    6120    7933#   7955
LOERAD= 177740            1954#   6600
LOWOCT  005466            3475#   5062#   5142#   7152#   7157    7314    7316    7319    7322    7327    7332    7335
LS      005510            3484#   5216#   5222#   6260
LSTCYL= 000632            4038#   6944    7422
LSTMEM  007303            4068#   6456
LSTTPK= 000002            4039#   6945    7423
LSTTST  014640            4765    4774#
LT      005722            3575#   4904    6265
MA      005742            3583#   4916    4918    6188    6189    6192    6193    6710    6721    6723    6728    6736    7359
                          7361    8433#   8434#   8435#   8642    8643
```

```
MAHILM  006106          3642* 4919  4922  6435* 6436* 6450* 6451* 6457  6677  6681  6683  6748  6751
                        7257  7358
MAPHOO= 170202          1952* 8823
MAPLOC= 170200          1951* 6659  8824
MAWRDC  010627          4207* 7339
MAXEQ   012003          4326* 7906
MAXHI1= 000000          4029* 5717
MAXHI2= 000000          4031* 5806  5826
MAXHI3= 000000          4033*
MAXHI4= 000001          4035* 5959  5977
MAXIH1  003210          3461* 5703  5708* 5902  7662* 7760*
MAXIH2  003220          3465* 5904  7666* 7757*
MAXIL1  003206          3460* 5706  5709* 5745  5901  7661* 7832
MAXIL2  003216          3464* 5903  7665* 7759* 7847
MAXLO1= 062031          4028* 5720
MAXLO2= 017500          4030* 5809  5829
MAXLO3= 112160          4032*
MAXLO4= 022370          4034* 5962  5980
MAXSEK  011663          4311* 5992
MCLK   = 000400          3072*
MDFLAG  003106          3406* 4396* 4405* 4408* 4516  4635  4815  5036  6904  6999  8357
MDFY15  011126          4243* 7216
MDS    = 001000          3016* 9706  10133 10508
MEMABT  003135          3429* 6610
MEMCSR= 172100          1953* 6574  6580  6621  6629
MEMSYS= 177744          1956* 6602
MEMVEC= 000114          3446* 4458* 4459*
MERD   = 001000          3073*
MESMSK= 000017          3066*
MEWD   = 002000          3074*
MICROS  012021          4330* 7897  7910  7923
MIDXFR  037032          8599* 8672
MIND   = 000200          3071*
MINEQ   011774          4324* 7893
MINHI1= 000000          4027* 5711
MINIH1  003204          3459* 5696  5701* 7660*
MINIH2  003214          3463* 7664*
MINIL1  003202          3458* 5699  5702* 5739  5803  5956  7659* 7829
MINIL2  003212          3462* 5823  5974  7663* 7844
MINLO1= 057467          4026* 5714
MMVEC  = 000250          1920* 4479* 4480*
MODP15  030654          5068  5147  7205*
MPERHD  026142          4458  6592*
MSP    = 000100          3070*
MXWRDC  031430          7338  7357*
M.CADD= 017760          3142*
M.CCIF= 017760          3141*
M.DPID  052462          5897  5905  7613  7801  7808  7883  10839*
M.DPIM  052422          7736  7756  7874  8283  8295  8305  10810*
M.DP01  052432          10813* 10822
M.DP02  052452          10817  10821*
M.DP40  052520          10850* 10871
M.DP41  052554          10855  10862*
M.DP42  052564          10861  10866*
M.DP44  052616          10873  10878*
M.DPSC  052630          10848  10882*
```

```
M.DRV = 000007           3138#
M.HEAD= 007000           3145#
M.ID  = 000003           3140#
M.PAR = 100000           3139#
M.SECT= 000760           3144#
M.SER = 077770           3143#
NCYL1   005512           3485#  5396*  5397   5401*  5402   5405*  5406   5411*  5412   5415*  5416   5418   5444*
                         5445   5451*  5452   5455*  5456   5461*  5462   5465*  5466   5468   5499*  5501   5507*
                         5509   5512*  5514   5875*  5877*  5879   5887
NCYL2   005514           3486#  5500*  5503   5508*  5513*
NED   = 010000           3019#  6336   9138   9148   9721   9799   10142  10512
NEDHDL  025112           6336#  6909
NEM   = 004000           3018#  9133   9736   9902   10142
NEWDRV  016772           5190#  5200   6299   8877
NEWMSG  007511           4094#  6477
NEWON   005530           3492#  6348*  6910*  6914
NEWPAS  016760           4401   5187#  6325
NGOFST  005546           3498#
NMTSTS= 000021           3566#  5254   5278   5304   5352   5391   5439   5495   5531   5555   5588   5638   5780
                         5861   5932   6009   6092   6167
NOCHK = 000400           3205#  10376  10438  10479  10506  10524  10545  10568  10577  10611
NOCLKS  011452           4286#  4509
NODRTS  007743           4124#  4625
NODSC = 004000           3210#  9359   9964
NOFALS  011443           4282#  4624   8515
NORTRY  003133           3427#  6850*  9416
NOTALD  010562           4213#  4902   4907
NOTOFT  016254           5024   5041#
NOTMSG  063526           4469   12325#
NOTYPE  003137           3432#  4467   4470*
NTREDY  007570           4104#  6930
NULINP  015054           4803   4818#
NXDRIV  007551           4101#  6919
NXF   = 000004           3030#
OCTBIN  051702           4848   5060   5140   5180   6492   7256   10642#
OFFSET= 000115           2958#  10421
OFINUS  005516           3487#
OFST  = 000004           3050#
ONECYL  011632           4306#  5840
ONLINE  005526           3491#
OPCOMP  003112           3410#
OPCONT  003121           3417#
OPI   = 020000           3042#  8489   9260   9742   9908   10144
OPIERR= 000010           4043#  9262   9411
OPNLST. 016144           4889   5017#  5022   5029
OR    = 000200           3014#
OREQSW= 000020           3437#
OVLODR  007333           4073#  4932
O.WAIT  003044           3354#  9642   9646*  9756*  9760   9776*  9842*  9855*  9870*  10414* 10474* 10536* 10553*
PACK  = 000103           2953#  6941
PARMDE  010543           4197#  4873
PARM0   002620           3243#  3398   6849   6977   8634   8647   9372   9437   9488   9519
PARM1   002704           3275#  6855*  9106   9472   9506
PAR   = 000000           1750#  1762   1765   1768   1778   1790   1798   1804   1805   1968   1970   1972   1974
                         1987   2009   3458   3466   3505   3525   3542   3547   3564   3573   3576   3581   3583
                         3588   3595   3598   3603   3605   3610   3620   3626   3636   3640   3648   3650   3659
```

```
                        3662    3667    3669    3674    4055    4058    4285    4365    4404    4464    4467    4492    4909
                        4948    5237    5243    6288    6331    6690    6781    6783    6968    7437    7515    8242    8411
                        8420    8761   12323   12324   12372   12406   12409
PART1   007273          4066#   4466
PART2   007277          4067#
PAT   = 000020          3068#
PATRN   005534          3494#   6175*   8044    8115    8222    8640
PATD0   006176          3680#   8046    8061    8117    8132
PATD1   006236          3702#
PATD2   006276          3724#
PATD3   006336          3746#
PATD4   006376          3768#
PATD5   006436          3790#
PATD6   006476          3811#
PATD7   006536          3833#
PATD8   006576          3854#
PATD9   006636          3876#
PATIO   006676          3898#
PATI1   006736          3920#
PATI2   006776          3942#
PATI3   007036          3964#
PATI4   007076          3986#   7973
PATI5   007136          4006#   7183    7261*   7267*
PBLKT   003102          3398#   9639    9726    9752    9920    9945   10363*
PBSVAL= 001000          3206#  10301
PCA   = 004000          3075#
PCD   = 010000          3076#
PCLKF   003122          3418#   4493*   4497*   7520    7541
PCVEC   003152          3445#   4498
PFIFTN  011040          4234#   4987
PGE   = 002000          3017#   9206   10142
PIP   = 020000          3060#
PIRQ  = 177772          1821#
PIRQVE= 000240          1915#
PKRB    003146          3443#
PKS     003142          3441#   4496    7528*   7545*
PKSB    003144          3442#   7524*   7527*
PLOFST  005544          3497#
PMA     005572          3505#   6192*   6193*   6643    6645    8063    8134    8214    8216    8614*   8615*   8616*   8617
                        8618    8638*   8639*   8642*   8643*
PROF_T  005756          3592#   4980    5031    5034    5073    5114
PREPAR  026362          6198    6641#   8623    8681
PREPKB  025330          4776    4961    4973    4998    5257    5281    5307    5355    5394    5442    5498    5534    5559
                        5591    5641    5783    5864    5935    6012    6095    6170    6401#   8388
PRMBUF  012432          4385#   7150*   7151
PRMINP  007370          4078#   4534
PRMLIM  006022          3615#   7314    7316    7324    7327    7329    7332
PRMLST  005712          3570#   4979    5032    5072    5113    7152    7156    7319*   7335*   7336*
PRMMEM  006132          3656#   4974    5053    5109    7150    7154    7210    7307
PRMPHC  005254          3471#   8821*  12058
PRMPLO  005252          3470#   8822*  12058
PRMPSP  012453          4389#   5094
PROMPT  012450          4388#   4657    4823    4874    5045    7233
PRVCMD  005222          3468#   4483    8810    9073
PRO   = 000000          1838#   4482    5190    6980    9438    9515
PR7   = 000040          1839#
```

```
PR2     = 000100          1840#
PR3     = 000140          1841#
PR4     = 000200          1842#  4475
PR5     = 000240          1843#  3330
PR6     = 000300          1844#
PR7     = 000340          1845#  4411   4459   4480   4500   6556   6598   11266  11267
PS      = 177776          1818#  1819   4411*  4482*  5190*  6980*  9628   9634*  9648*  10357  10358*  10617*
PSTART  012530            1971   4408#
PSW     = 177776          1819#  9438*  9515*
PT      005740            3582#  4982   6175   6176   6202   6243   6708   7213
PWRMSG  054324           11269  11273#
P.WRVEC= 000024            1910#  4426*  4427*  11257*  11265*  11266*
P.ASOF= 000032            3227#  9060   9970*  9971   10004*  10235*  10423*  10424
P.AOO   = 000040          3230#  6925   6933   9063   9478   9961*  9962   9978*  9979  10292*  10500*
P.A01   = 000044          3232#  9065   10271*
P.A10   = 000050          3234#  9067   10278*
P.A11   = 000054          3236#  7387   9069   10285*
P.BAHI= 000007            3189#  6191*  6667*  8620*  9049
P.BALO= 000010            3190#  6020*  6100*  6179*  6188*  6666*  6946*  7424*  8407*  8457*  8469*  8617*  9052
                         10457  10554
P.BAR   = 000024          3224#  9059   10001*  10231*
P.BOO   = 000042          3231#  9064   9956*  9957   10293*  10501*
P.B01   = 000046          3233#  9066   10272*
P.B10   = 000052          3235#  9068   9500   10279*
P.B11   = 000056          3237#  9070   9877*  10286*
P.CMND= 000001            3183#  5206*  5259*  5653*  5791*  5872*  5943*  6013*  6042*  6045*  6096*  6121*  6221*
                          6225*  6233*  6240*  6852*  6908*  6941*  6943*  6961*  6978*  7383*  7421*  7449*  7452*
                          7686*  7703*  7956*  8371*  8410*  8413*  8448*  9040   9385*  9428*  9473*  9514*  9772
                         10389  10393  10405  10415  10421  10427  10430  10436  10461  10468  10487  10534  10551
                         10573  10582
P.CS1   = 000016          3221#  5661*  5683*  5684   6871   9053   9181   9199   9538   9571   9793*  9845*  9883*
                          9884*  9886*  9887   9996*  9998*  10225*  10228*  10379  10433*  10434*  10436*  10440*  10441
                         10465*  10468*  10470*  10471*  10481*  10482  10490*  10491*  10493*  10494  10496*  10497  10502
                         10541*  10542*  10544*  10547*  10548  10564*  10565*  10567*  10570  10589*  10590  10599*  10600
P.CS1H= 000007            3188#  5208   5210*  5212*  5652*  5660*  5682*  6856*  6948*  7417*  8447*  9511*  9513*
                          9517*  9518*  9883   10124  10433  10470  10490  10541  10555  10564
P.CS2   = 000020          3222#  6336   9054   9128   9133   9138   9140   9146   9148   9206   9256   9340   9799
                          9999*  10229*  10387*  10412  10477*  10478  10489  10508  10512  10538*  10539  10563
P.CYLN= 000002            3184#  5262*  5283*  5286*  5309*  5312*  5314*  5315   5318*  5320   5357*  5360*  5364*
                          5368*  5369*  5373*  5397*  5418*  5445*  5447*  5468*  5470*  5501*  5503*  5536*  5559*
                          5561*  5563*  5597*  5599*  5615*  5617*  5792*  5795*  5812   5815*  5832   5873*  5879*
                          5883*  5944*  5948*  5966*  6014*  6097*  6180*  6200*  6254*  6255   6270*  6271   6274*
                          6275   6944*  7422*  7455   7458   8023   8254   8259   8288   8300   8311   8449*  8551
                          8605*  9042   9508*  9881   10417  10459  10561
P.DCYL= 000030            3226#  8555   8601   9055   9313   9528   9560   10003*  10237*
P.DRVN= 000000            3182#  6854*  6899*  9769   9946   10122  10361
P.DS    = 000036          3229#  9061   9191   9195   9246   10006*  10233*  10514
P.DTS   = 000026          3225#  8557   8559   8602   8603   9056   9314   9316   9499   9531   9532   9561   9567
                         10002*  10232*
P.EPAT= 000062            3239#  9337   9795*  10381
P.EPOS= 000060            3238#  9336   9796*
P.ER    = 000034          3228#  8489   9062   9127   10005*  10234*
P.OFST= 000006            3187#  10423
P.PRST= 000014            3192#  9154   9167   9349   9354   9359   9364   9475   9641*  9754*  9788*  9816*  9823
                          9827*  9924*  9947   9951*  9964*  9982*  9992*  10018  10022*  10032*  10044*  10149*  10301*
                         10376*  10413*  10437*  10438  10463  10469*  10475  10479  10506  10516*  10524  10545  10568
```

```
                    10577  10611
P.SECT= 000004       3185#  6019*  6028   6032*  6033   6037*  6039   6050   6074*  6075   6099*  6183*  6208*
                     6246*  6247   6259*  6260   7420*  795C   7961*  8000   8278   8454*  8463*  8464   8470*
                     8480*  8481   8553   8607*  9045   9507*  9882   10418  10460  10562
P.TRCK= 000005       3186#  6015*  6098*  6103   6107*  6108   6117*  6118   6124*  6125   6181*  6207*  6250*
                     6251   6262   6264*  6945*  7423*  7951*  7953   7958*  7959   8014   8261   8291   8450*
                     8552   8606*  9043
P.WC  = 000012       3191#  6021*  6101*  6182*  6209*  6210*  6947*  7425*  8319   8332*  8408*  8425   8456*
                     8610*  8636   8673   9047   10458
P.WCR = 000022       3223#  9057   10000*  10230*
ONEWSW= 000000       2942
Q.INIT= 040000       3213#
RCDASW= 000010       3436#
RCLREQ= 004000       4051#  9184   9189   9234   9239   9244   9249   9383
RDALHD= 000164       2971#  9772   10551
RDCHR = 104406       7045   11568#
RDCHRS  030174       4592   4644   4659   4754   4798   4825   4875   4933   5019   5047   5096   5168   6478
                     7030#  7217   7235
RDDATA= 000121       2960#  6045   6233   6943   7421   8448
RDGATE= 100000       3079#
RDHDO   043750       9271   9296   9497#
RDHEAD= 000125       2962#  5206   5653   7452   9514   9886   10567
RDSTAT= 000141       2970#  6908   7383   9473   10487
RDY   = 000200       2985#  9379   10444  10497
RECAL = 000113       2957#  5259   6961   8371   9385   10430
RECODE  005474       3478#  6051   6950   6975*  8177   8666   8669   8747*  8755*  8846   8856*  9103*  9104
                     9131*  9136*  9157   9159   9184*  9189*  9229*  9234*  9239*  9244*  9249*  9254*  9262*
                     9266   9272   9287*  9291   9297   9312*  9319   9322*  9326*  9329*  9342*  9343*  9352*
                     9357*  9362*  9367*  9375   9381*  9383   9389   9391*  9411   9413*  9418   9425*  9445*
                     9477*
REDBSF  036242       5225   8445#
REISSU  003130       3424#  6218*  9002*
RELEAS= 000140       2969#  6240   10534
REPLPK  007654       4114#  4642
REPSUP  041074       5670   6551   6595   7457   7708   8114   8466   9037#  9109
RESREG= 104410       5674   5723   6462   6543   6668   6713   6794   7348   7406   7619   7741   7761   7911
                     7938   7962   7979   8026   8095   8176   8237   8334   8338   8415   8436   8541   8565
                     8624   8652   8748   8833   8973   9088   9440   9489   9520   9579   10914  10966  11570#
RESVEC= 000010       1905#
RETANL  043552       9107   9386   9444#
RETNML  043570       6351   8493   9108   9447#
REVRSE  011744       4320#  5914   7843
RHEDHD  021744       5658   5760#
RKASOF= 000016       2938#  9466   9782   9852   9932   10187  10235  10424*  10449
RKBA  = 000004       2933#  9465   10183  10231  10457*
RKBADP  007422       4083#  4538
RKBAS   003026       3328#  4519*  4523*  4537   4540*  6848   7706   9274   9299   9378   9427   9458   9635
                     9704   10368
RKCS1 = 000000       2931#  4653*  5684*  5761   6921*  7689   7707*  9174*  9275*  9279   9300*  9304   9460
                     9727   9779   9837   9887*  9896   10078*  10079  10088*  10127*  10128  10152*  10180  10441*
                    10443   10482*  10494*  10496  10526*  10548*  10570*  10579*  10589  10599  10614*
RKCS2 = 000010       2935#  9461   9705   9865   10123*  10181  10229  10412*  10478*  10489*  10539*  10563*  10588*
RKDA  = 000006       2934#  9463   9882*  10184  10232  10418*  10460*  10562*
RKDB  = 000024       2941#  5217   5218   7454   9276   9277   9278   9301   9302   9303   9961   9862   9863
RKDC  = 000020       2939#  9462
RKDCYL= 000020       2940#  9881*  10188  10237  10417*  10459*  10561*
```

```
RKDS   = 000012          2936#   9379    9467    9730   10185   10233
RKECPS= 000030          2946#   9796   10192
RKECPT= 000032          2948#   9795   10193
RKER   = 000014          2937#   9468    9739    9905   10186   10234
RKMR1  = 000026          2942#   9873#   9952#  10189   10266#  10273#  10280#  10287#  10388#
RKMR2  = 000034          2943#  10190   10500
RKMR3  = 000036          2944#  10191   10501
RKPAT  = 000032          2947#
RKPOS  = 000030          2945#
RKPRI    003032          3330#   4478    4528#   4547    4563#   9634   10358
RKPRTY   007462          4089#   4552
RKVADR   007442          4086#   4543
RKVEC    003030          3329#   4476    4524#   4525#   4542    4545#   5658#   5754#   7685#   7700#
RKWC   = 000002          2932#   9464   10182   10230   10459#
RLS    = 000010          3009#  10538
RNOADR   032122          5535    7505#
RNOLNG= 016514          4037#
RNOSHT= 002734          4036#
ROTIMS   011552          4297#   5738
RUNTST   0:6664          4929    5165#
RVTERM   033126          5886    7750#
RWBUF    063526          3583    3605    3640    6020    6047    6100    6179    6608    6946    6953    7424    7429    7934
                         7936    8048    8119    8401    8407    8638   12323#
RWDTST   024040          5239    6130#
RWTINX= 000040          6131#   6706
R.ABNL   047740          9647    9757    9832    9849   10027   10047   10307#  10517
R.CONT   047764          9709    9747    9811    9820    9891    9913    9936    9942    9975   10015   10039   10084   10136
                        10159   10297   10315#  10531   10585   10594   10607
R.NORM   047752          9785    9878    9987   10311#  10454   10527   10615
SAVPAR   003176          3456#   6655#   8067    8138
SAVPRS   005550          3499#   8635    8646
SAVREG= 104407          5688    6430    6529    6641    6701    6773    6776    7304    7381    7564    7730    7750    7794
                         7933    7949    7971    7994    8042    8109    8171    8256    8399    8423    8503    8550    8599
                         8633    8645    8737    8809    8868    8986    9038    9100    9457    9498    9559   10899   10940
                        11569#
SAVWRD   003200          3457#
SAVXOP   027032          6740    6773#
SCLR   = 000040          3012#  10588
SCNDRV   027420          4723    6898#
SCOPER= 104411          6242   11571#
SCOPE1   054346         11284#  11571
SCRACH   005532          3493#   4614#   4615#   4616    5595#   5601#   5613#   5619#   5680#   5725    5876#   5878#   5946#
                         5983#   5984    6049#   6055    6060#   6063    7080#   7082#   7083    7093#   7094#   7095    7734
                         7754    8113#   8179    8192#   8195
SECNL1   010651          4211#   4897
SECNL2   010701          4216#   4901
SEEK   = 000117          2959#   5791    5872    5943    7449    7686   10405   10415
SEEKER   031742          5263    5284    5287    5310    5313    5319    5358    5361    5365    5370    5374    5398    5419
                         5446    5448    5469    5471    5502    5504    5537    5560    5562    5564    5598    5600    5616
                         5618    7448#
SEEKHD   033040          7685    7717#
SELDRV= 000101          2952#  10544
SELECT   005524          3490#
SELP15   011076          4239#   7229
SERNM    011363          4273#   7386    7428
SETPRM   016430          4892    5093#   5099    5104    5148
```

```
SETTUP  030032          5252    5276    5302    5350    5389    5437    5493    5529    5553    5586    5636    5778    5859
                        5930    6007    6036    6090    6112    6116    6165    6215    6973#
SIZMEM  025362          4511    6430#
SKI   = 000002          3028#   9231
SM      005754          3588#   5593
SOFTBS  011411          4277#   8539
SPACE1  012430          4383#   4822    5044    7124    7232
SPACE2  012427          4382#   8520    8526    8915    8919    8922    8925    8945    8950    8965
SPACE3  012426          4381#
SPACE4  012425          4380#
SPACE6  012423          4379#   7131
SPAR  = 020000          2992#   9181   10140
SPCDMX  012234          4355#   5912    5917
SPDLSS= 000020          3052#   9195
SRETBS  037414          8698#   8739
SRTSPL= 000111          2956#  10427
SR0   = 177572          1924#   6542*   6552    6818*   6820*   6822*   6824*   8068*   8083*   8139*   8235*   11439   11454#
                       11461#
SR1   = 177574          1925#   6553
SR2   = 177576          1926#   6554
SR3   = 172516          1927#   6283*   6541*   6557   11451*
ST      005752          3587#   5223    5623
STACK = 001100          1809#   4418    4588    5191    6973*   6974    8370   11268
STALL   032040          7476#   8781
STALLS  005502          3481#   4488*   4589*   4871*   5223*   5596*   5614*   5623*   7485
STKLMT= 177774          1820#
STPASS  016746          5039    5185#
STRCMD  037776          5663    7690    8785    8809#
SUBCLR= 000177          2973#   6852    6978    7703    9428   10582
SUBVER  007216          4058#   4465
SUMHI1  003170          3453#   5692*   5693*   5731*   5799*   5800*   5892    5899*   5910    5952*   5953*   7341*   7567*
                        7589*   7590*   7598*   7611    7656*   7740*   7798    7802*   8258*   8308
SUMHI2  003174          3455#   5819*   5820*   5894    5907*   5915    5970*   5971*   7658*   7737*   7806    7809*
SUMLO1  003166          3452#   5691*   5732*   5751    5798*   5891    5898*   5951*   7340*   7342    7566*   7588*   7599*
                        7612    7655*   7799    7803*   7839    8257*   8284*   8296*   8306
SUMLO2  003172          3454#   5818*   5893    5906*   5969*   7657*   7739*   7807    7810*   7854
SVAL  = 100000          3062#
SVPRMS  037172          6216    8633#   8678
SWR     001140          2039#   4416    4438*   4440    4446*   4453*   6061    6069    6172    6219    6223    6229    6231
                        6235    6509    7450    7478    8193    8220    8878    8905    8933   11174   11181   11193   11196
                       11286   11309   11327   11334
SWREG   000176          1965#   4446    6509    6511    6514*
SWRMSG  007502          4092#   6512
SW0   = 000001          1873#
SW00  = 000001          1863#   1873
SW01  = 000002          1862#   1872
SW02  = 000004          1861#   1871
SW03  = 000010          1860#   1870
SW04  = 000020          1859#   1869
SW05  = 000040          1858#   1868
SW06  = 000100          1857#   1867
SW07  = 000200          1856#   1866
SW08  = 000400          1855#   1865
SW09  = 001000          1854#   1864
SW1   = 000002          1872#
SW10  = 002000          1853#
```

```
SW11   = 004000          1852#
SW12   = 010000          1851#
SW13   = 020000          1850#   8878
SW14   = 040000          1849#
SW15   = 100000          1848#
SW2    = 000004          1871#
SW3    = 000010          1870#
SW4    = 000020          1869#
SW5    = 000040          1868#
SW6    = 000100          1867#
SW7    = 000200          1866#
SW8    = 000400          1865#
SW9    = 001000          1864#
S.ACLO= 000100           3097#
S.BRHM= 000100           3112#
S.BRKE= 040000           3134#
S.CART= 000400           3114#
S.DCLO= 010000           3104#
S.DIB  = 002000          3130#
S.DOOR= 000200           3113#
S.DRA  = 000040          3083#
S.DRCT= 020000           3105#
S.DRY  = 000200          3085#   6925
S.DSC  = 040000          3092#   9814    9962    9979    10042
S.FLT  = 000200          3098#   9957
S.FORM= 001000           3087#
S.FWD  = 002000          3116#
S.HDFL= 000200           3127#
S.HDHM= 000040           3111#
S.ICYL= 000040           3096#
S.ILF  = 000400          3099#
S.LIMO= 020000           3133#
S.LOAD= 010000           3118#
S.MHC  = 000400          3128#
S.NMCV= 010000           3132#
S.OFF  = 002000          3088#
S.PAR  = 001000          3100#   9817    10294
S.PIP  = 020000          3091#   9825    10020
S.PLO  = 004000          3131#
S.REV  = 004000          3117#
S.RTZ  = 020000          3119#
S.SECT= 000020           3124#
S.SKI  = 002000          3101#
S.SPIN= 010000           3090#
S.SPLS= 010000           3103#
S.SPOK= 001000           3115#
S.TYPE= 000400           3086#
S.UNLO= 040000           3120#
S.UNS  = 040000          3106#
S.VV   = 000100          3084#
S.WCLK= 000040           3125#
S.WGAT= 000100           3126#
S.WLE  = 004000          3102#
S.WRL  = 004000          3089#   6933
S.XCOK= 000020           3110#
S.XERR= 001000           3129#
```

```
SO       005726          3577#   4895    5221
S1       005730          3578#   4895    5222
S2       005732          3579#   4899    5215
S3       005734          3580#   4899    5216
TBITVE=  000014          1906#
TCONHI   003156          3448#   7614*   7872
TCONLO   003154          3447#   7616*   7873
TELDRV   013744          4597    4605#   4662    4673    4717    4730
TIMER    032516          5685    7585    7635#   7640    7693
TIMSEK   032660          5796    5816    5880    5884    5949    5967    7681#
TIMSKP   011466          4288#   7577
TKVEC =  000060          1913#   4474*   4475*
TLSTHD   010365          4176#   4774    4819
TOPROC   043600          5671    7709    8989    9156    9456#
TPVEC =  000064          1914#
TRACK    005520          3488#
TRAPVE=  000034          1912#   4424*   4425*   11267*
TRKCHK   033674          6113    6123    7949#
TRKNLW   010713          4218#   4906
TRNSFR   037304          6222    6226    6234    8662#
TRTVEC=  000014          1907#
TSTDRN   011320          4265#   5228
TSTING   003110          3408#   4481*   5186*   5649*   5655*   8353
TSTLST   005606          3524#   3566    4790    4811    4833    4853*   4858    4862*   6705    7008    7132
TSTMDS   010161          4150#   4752
TSTMSG   055560          8899    11573#
TST1     017316          5238    5250#
TST10    020542          5495    5527#
TST11    020614          5531    5551#
TST12    020702          5555    5584#
TST13    021142          5588    5634#
TST14    021762          5638    5647    5776#
TST15    022302          5780    5789    5797    5817    5857#
TST16    022640          5861    5870    5881    5885    5928#
TST17    023164          5932    5941    5950    5968    6005#
TST2     017376          5254    5274#
TST20    023604          6009    6088#
TST21    024040          6092    6163#
TST3     017456          5278    5300#
TST4     017564          5304    5348#
TST5     017730          5352    5387#
TST6     020134          5391    5435#
TST7     020364          5439    5491#
TWOTOS=  002000          4050#   9157    9477
TYPAVC   033626          5752    5911    5916    7840    7855    7919#
TYPBSF   036530          5235    8503#
TYPDS =  104405          5742    5748    6314    7835    7850    11565#
TYPE  =  104401          4464    4465    4466    4469    4473    4509    4534    4538    4543    4552    4564    4566    4567
                         4591    4601    4602    4605    4613    4616    4624    4625    4634    4642    4652    4657    4671
                         4672    4752    4753    4760    4761    4774    4778    4797    4806    4807    4819    4822    4823
                         4865    4866    4870    4873    4874    4881    4882    4897    4901    4902    4906    4907    4924
                         4932    4939    4940    4963    4987    5000    5018    5027    5028    5044    5045    5076    5077
                         5094    5102    5103    5167    5176    5177    5228    5738    5743    5744    5749    5750    5753
                         5840    5889    5909    5912    5914    5917    5992    6312    6315    6369    6375    6379    6383
                         6385    6386    6417    6418    6456    6460    6461    6477    6487    6488    6512    6906    6918
                         6919    6929    6930    6937    6938    6952    6957    6958    7050    7056    7062    7068    7079
```

```
                    7083    7087    7095    7100    7102    7103    7105    7124    7131    7151    7176    7182    7216
                    7225    7226    7229    7232    7233    7271    7272    7339    7345    7385    7386    7405    7427
                    7428    7432    7576    7577    7828    7831    7837    7841    7843    7846    7852    7856    7893
                    7897    7906    7910    7919    7923    8381    8382    8507    8508    8509    8510    8515    8516
                    8520    8526    8530    8539    8876    8892    8893    8894    8898    8899    8904    8907    8908
                    8909    8910    8915    8918    8919    8920    8921    8922    8925    8928    8929    8931    8945
                    8950    8954    8960    8965    8966    8968   10760   11007   11072   11147   11176   11184   11269
                   11561#
TYPERR  040166      8868#  11183
TYPFMT  003114      3412#
TYPLST  015672      4886    4959#
TYPMAX  033602      5746    7833    7848    7906#
TYPMIN  033556      5740    7830    7845    7893#
TYPOC = 104402      8519    8525    8529    8914    8924    8927    8947   11562#
TYPON = 104404     11564#
TYPOS = 104403      6474    7128    7133    7179    7184    8901   11563#
TYPPAT  030622      4999    7175#   7231
TYPPRM  030542      4962    5043    7150#
TYPTMS  033336      5842    5994    7826#
TYPTST  030506      4777    4821    7124#
T.ASOF  003004      3315#   9782*   9783    9807    9852*   9853    9932*   9933    9939    9970   10004   10012   10036
                   10187*  10449*  10450
T.BA    002776      3312#  10001   10183*
T.CS1   002770      3307#   9727*   9779*   9780    9793    9804*   9837*   9839    9845    9874*   9896*   9898    9928
                    9953*   9967*   9997*   9998   10009*  10033*  10079*  10080   10120   10124*  10125*  10127   10131
                   10140   10180*  10227*  10228   10268*  10275*  10282*  10289*  10443*  10444   10446
T.CS2   002772      3309#   9705*   9706    9721    9724    9736    9765    9865*   9856    9902    9916    9999   10121*
                   10122*  10123   10133   10142   10181*
T.DA    003000      3313#  10002   10184*
T.DB    003024      3324#
T.DC    003002      3314#  10003   10188*
T.DS    003010      3318#   9730*   9731   10006   10185*
T.ER    002006      3317#   9739*   9742    9905*   9908   10005   10144   10186*
T.MR1   003012      3319#  10189*
T.MR2   003014      3320#   9814    9825    9961    9978   10020   10042   10190*  10271   10278   10285   10292
T.MR3   003016      3321#   9817    9877    9956   10191*  10272   10279   10286   10293   10294
T.PAT   003022      3323#  10193*
T.POS   003020      3322#  10192*
T.WCR   002774      3311#  10000   10182*
UBMPRS  003134      3428#   6281    6454*   6539    6656    8181    8819
UEXATT= 000010      3199#   9364    9788
UFE   = 000400      3015#   9140    9146    9721    9799   10142   10512
UNLOAD= 000107      2955#  10393
UNS   = 040000      3043#   9226
UPE   = 020000      3020#   9128    9736    9902   10142
VFHDSW= 000002      3434#
VV    = 000100      3055#
WC      005746      3585#   4910    6209    6715
WCDASW= 000004      3435#
WCE   = 040000      3021#   9340    9736    9902   10142
WCEFLG  003131      3425#   6227    8459*   8461    8476*   8478    8492*   8664*   8668*
WCERR = 000100      4046#   8666    9343    9411
WC2BIG  010725      4220#   4924
WDSXFR  005562      3500#   8043    8112    8168    8565*   8608    8636*   8637*
WHDSW = 000001      3433#
WLE   = 004000      3040#   9222    9742    9908
```

```
WORDSP  012436          4386*  7176
WRDATA= 000123          2961*  6013   6096   6121   6221   8410   10465
WRHEAD= 000127          2963*
WRL   = 004000          3059*
WRTCHK= 000131          2964*  6042   6225   7956   8413   10461
WRTGAT= 040000          3078*
WRTLOK  007604          4106*  6938
WRTSEC  036110          8399*
W.DRV   003104          3404*  9175*  9636*  9751*  9777*  9829*  9844*  9857*  9871*  9919*  9986*  9995*  10024*
                       10046*  10317* 10366* 10429* 10432* 10453* 10505* 10523*
W.MILI  003050          3357*  9631   10597
W.MIN   003056          3373*  9175   10429
W.MTIM  003046          3356*  9629*  9631*
W.SEC   003052          3370*  10366
W.TIME  003070          3379*  9178*  9632   9638*  9750*  9778*  9828*  9843*  9856*  9872*  9918*  9985*  9994*
                       10023*  10045* 10316* 10365* 10452* 10504* 10522*
W.WCK = 000200          3204*  10376  10437  10463  10469
W.WTCH  044356          8789   8799   9624*  10442  10448  10495
W.8SEC  003054          3372*  9829   10024  10432
XDPSAV  005540          3496*  6738*  6739*  6753*  6754*  6785   6786   6812
XDPSVD  003125          3421*  4512   4515*  6724*  6734*  6757*  6778   8368*
XDP1    027060          6775   6781*
XDP2    027126          6779   6782   6794*  6801   6832
XDP3    027152          6788   6803*
XDP4    027132          6790   6796*
XOVLAD  003124          3420*  6171*  6194*  6278   6777*  8362
XXDPAD  005536          3495*  6679*  6683*  6684*  6685*  6726   6732   6784
XXDPCH  003107          3407*
$APTHD  001000          1996   2002*
$ASTAT= ****** U       11385  11400
$ATYC   054654         11356  11358*
$ATY1   054630         11354*
$ATY3   054636         10745  11355*
$ATY4   054646         11189  11357*
$AUTOB  001134          2036*
$BASE   001374          2143*  4519
$BOADR  001122          2031*
$BDDAT  001126          2033*
$BELL   001310          2094*  11176  11208
$CHARC  052416         10762* 10772* 10779  10788* 10793*
$CKSWR= ****** U       11568
$CMTAG  001100          2019*  4413   4414   4422   4428   4429   4430
$CM1  = 000040          2051*  2052*  2053*  2054*  2055*  2056*  2057*  2058*  2059*  2060*  2061*  2062*  2063*
                        2064*  2065*  2066*  2067*  2068*  2069*  2070*  2071*  2072*  2073*  2074*  2075*  2076*
                        2077*  2078*  2079*  2080*  2081*  2082*  2083*
$CM2  = 000100          2051*  2052*  2053*  2054*  2055*  2056*  2057*  2058*  2059*  2060*  2061*  2062*  2063*
                        2064*  2065*  2066*  2067*  2068*  2069*  2070*  2071*  2072*  2073*  2074*  2075*  2076*
                        2077*  2078*  2079*  2080*  2081*  2082*  2083*
$CM3  = 000040          2049*  2051
$CM4  = 000011          2083*  2084*  2085*  2086*  2087*  2088*  2089*  2090*  2091*  2092*
$CNTLG  054207         11246*
$CNTL   054202         11245*
$CORE   055304         11437  11464*
$CPUOP  001346          2117*
$CRLF   001315          2096*  4564   4634   4778   4963   5000   5753   6386   6460   6461   7100   7105   7345
                        7405   7432   7831   7841   7846   7856   8510   8516   8530   8898   8908   8910   8918
```

```
                      8921    8931    8954    8960    8968   10761   10796   11184   11208
$CROUT  055334       11464   11471*
$DBLK   053656       11113   11147   11155*
$DB2D   052760        7895    7908    7921   10940*
$DB20   052640        6458    7158    7343    7430   10899*
$DECVL  053140       10942   10988*
$DEVCT  001330        2108*   5188*   6298*
$DEVM   001376        2144*   4702
$DOAGN  025066        6308    6317    6323*
$DTBL   053646       11116   11151*
$ENDAD  025056        1982    6319*  11203
$ENDCT  025024        4428    6310*
$ENDMG  025075        6312    6327*
$ENULL  025072        6315    6326*
$ENV    001340        2113*  10740   11186   11363   11387
$ENVM   001341        2114*   4451    4520    4703    4715   10742   10747   11365
$EOP    024752        6167    6296*
$EOPCT  025016        4428*   5038*   5182*   6307*   6311
$ERFLG  001103        2022*  11171*  11208   11284   11298   11323   11325   11331*  11351
$ERMAX  001115        2028*   4431*   4461*   6996*  11325   11346*  11351
$ERROR  053666        4422   11170*
$ERRPC  001116        2029*  11178*  11179*  11180   11208   12043
$ERRTB  001400        2161*   8889
$ERTTL  001112        2026*  11177*  11208
$ESCAP  001306        2093*   4430*  11199   11201   11208   11345*
$ETABL  001340        2112*
$ETEND  001400        2008    2145*
$FATAL  001322        2105*  11391*
$FFLG   055074       11354*  11357*  11385   11394*  11402*
$FILLC  001156        2047*  10765   10796
$FILLS  001155        2046*  10796
$GDADR  001120        2030*
$GDDAT  001124        2032*
$GET42  025046        6316*
$GTSWR= ****** U     11567
$HD   = 000000        1778
$HIBTS  001000        2003*
$HINUM  052134        4490*   7976   10697   10705   10710*  10715*
$HIOCT  052034        5063    5143    6495    7259   10667*  10680*
$ICNT   001104        2023*  11336*  11337   11339*  11350
$INTAG  001135        2037*
$ITEMB  001114        2027*   8884   11180*  11188   11208
$KTNEX  055276       11438   11463*
$KTOUT  055266       11455   11460*
$KT11   055134        6195    6431*   6433    6787    8064    8135    8224    8233    8621    8679   11436*  11440*  11463*
$LF     001316        2097*  10796   11208
$LFLG   055073       11395*  11401*
$LONUM  052136        4491*   7481    7506    7975   10696   10703   10709*  10716*
$LPADR  001106        2024*   4432*  11271   11329*  11343*  11348   11350
$LPERR  001110        2025*   4433*   6035*   6111*   6115*   6214*  11198   11288   11329   11344*  11350
$LSTAD  055400        6435    6448   11476*  11483*
$LSTBK  055402        6439   11477*  11484*
$MADR1  001352        2130*
$MADR2  001356        2134*
$MADR3  001362        2137*
$MADR4  001366        2140*
```

```
$MAIL   001320          2004    2008    2103*   4450    5251    5275    5301    5349    5388    5436    5492    5528    5552
                        5585    5635    5777    5858    5929    6006    6089    6164    10740   11186   11342
$MAMS1  001350          2124*
$MAMS2  001354          2132*
$MAMS3  001360          2135*
$MAMS4  001364          2138*
$MBADR  001002          2004*
$MFLG   055072         11355*  11361   11396*  11400*
$MNEW   054225         11249*
$MSGAD  001334          2110*  11371*  11374
$MSGLG  001336          2111*  11376*
$MSGTY  001320          2104*  11369   11377*  11389   11393*
$MSWR   054214         11247*
$MTYP1  001351          2125*
$MTYP2  001355          2133*
$MTYP3  001361          2136*
$MTYP4  001365          2139*
$MXCNT  054626         11340   11350*
$NULL   001154          2045*  10767   10796
$NWTST= 000001          5243*   5245    5267*   5269    5291*   5293    5325*   5327    5378*   5380    5424*   5426    5476*
                        5478    5520*   5522    5541*   5543    5568*   5570    5627*   5629    5768*   5770    5846*   5848
                        5921*   5923    5998*   6000    6080*   6082    6132*   6134
$OCNT   053436         11044*  11073*  11086*
$OCTVL  052742          7388   10901   10926*
$OMODE  053440         11039*  11043*  11048   11051*  11062*  11088*
$OVER   054612         11310   11330   11338   11347*
$PASS   001326          2107*   4450*   5185*   5229    6304*   6305*   6313    6326    7031
$PASTM  001006          2006*
$PWRCT  054322         11262*  11263*  11272*
$PWRDN  054236          4426   11257*  11265
$PWRUP  054250         11257   11262*
$QUES   001314          2095*   4567    4602    4672    4761    4807    4866    4882    4940    5028    5077    5103    5177
                        6418    6488    7103    7226    7272   10796   11208
$RAND   052036          7480    7505    7974   10692*
$RDCHR  054062         11224*  11568
$RDDEC= ****** U       11569
$RDLIN= ****** U       11569
$RDOCT= ****** U       11569
$RDSZ = 000000         11245*
$REGAD  001160          2049*
$REG0   001162          2051*   9041   12043
$REG1   001164          2052*  12043
$REG10  001202          2059*   6065*   6557*   6602*   8167*   8168*   8169    8207*   8210    9276*   9301*   9336*  12047
                       12055
$REG11  001204          2060*   6066*   8208*   9277*   9302*   9337*  12047   12055
$REG12  001206          2061*   6067*   8209*   9278*   9303*   9330*   9332*  12047   12055
$REG13  001210          2062*   8211*   8213*   8215*   8216*  12047   12055
$REG14  001212          2063*   8210*   8212*   8214*  12050   12055
$REG15  001214          2064*   8217*   8218*  12050   12055
$REG16  001216          2065*   8146*  12051   12055
$REG17  001220          2066*  12051
$REG2   001166          2053*  12043
$REG20  001222          2067*  12051
$REG21  001224          2068*  12051
$REG22  001226          2069*  12051
$REG23  001230          2070*  12051
```

```
$REG24  001232          2071*  12051
$REG25  001234          2072*  12051
$REG26  001236          2073*   8912    9072
$REG27  001240          2074*
$REG3   001170          2054*  12043
$REG30  001242          2075*
$REG31  001244          2076*
$REG32  001246          2077*
$REG33  001250          2078*
$REG34  001252          2079*
$REG35  001254          2080*
$REG36  001256          2081*   9051*  12046
$REG37  001260          2082*   9052*  12046
$REG4   001172          2055*
$REG5   001174          2056*   6016*   6552*   6596*   7458*   8183*   8189*   8197    8203*   8852*   8853*   9048    9422*
                        9423*   9459    9528*   9548*   9560*   9574    9577   12047   12054
$REG6   001176          2057*   6017*   6553*   6600*   6608    6624*   7459*   8184*   8190*   8199    8204*   9529*   9532*
                        9543    9545*   9547*   9567*   9568*   9569*   9570*   9573*   9575    9576   12047   12054
$REG7   001200          2058*   6018*   6050*   6554*   6601*   6606    6622*   8191*   8201    8205*   9530*   9531*   9533
                        9536*   9540*   9542*   9574*   9576*   9578*  12047   12054
$RESRE  055442         11518*  11570
$FTNAD  025070          6325*
$R2A   = ****** U      11571
$SAVRE  055404         11502*  11569
$SCOPE  054372          4420   11306*
$SETUP= 000037          4392*   4419    4420    4422    4424    4426    4428    4429    4430    4432    6302   11171   11196
                       11203   11213   11251   11307
$SIZE   055076          6432   11424*
$SIZEX  055340         11462   11472*
$STUP  = 177777         4392*
$SUPRS  053154          6459    7159    7344    7404    7431    7896    7909    7922   10999*
$SVLAD  054556         11318   11341*
$SVPC  = 000230         1980*   1985
$SWR   = 167000         1758*   1778    1782    1783    1784    1785    1786    1787    1788    1789    2092    2093    2094
                        4429    4430    4432    4433    5251    5275    5301    5349    5388    5436    5492    5528    5552
                        5585    5635    5777    5858    5929    6006    6089    6164    6293    6303    6318    6324    6326
                       11162   11163   11164   11165   11166   11174   11181   11193   11196   11208   11299   11300   11301
                       11302   11303   11309   11321   11323   11324   11325   11332   11333   11334   11344   11347   11350
$SWREG  001342          2115*   4453
$SWRMK= 000000         11303
$TESTN  001324          2106*   5251*   5275*   5301*   5349*   5388*   5436*   5492*   5528*   5552*   5585*   5635*   5777*
                        5858*   5929*   6006*   6089*   6164*  11342*
$TIMES  001304          2092*   4429*   5202*   6303*   7003*   7008*   8367*  11307   11332*  11337   11340*  11350
$TKB    001146          2042*   6359    6401*  11211   11228   11234
$TKS    001144          2041*   4793*   4984*   5004*   6301*   6370*   6403*  11211   11226   11232
$TMPU   001262          2083*
$TMP1   001264          2084*   8702*   8703*   8711
$TMP10  001302          2091*   8715*   8716*   8717
$TMP2   001266          2085*   8698*   8726
$TMP3   001270          2086*   8699*   8727
$TMP4   001272          2087*   8700*   8728
$TMP5   001274          2088*
$TMP6   001276          2089*   5593*   5596    5604*   5612*   5614
$TMP7   001300          2090*   5605*   5611*
$TN    = 000022         1767*   1778    5243    5251*   5254    5267    5275*   5278    5291    5301*   5304    5325    5349*
                        5352    5378    5388*   5391    5424    5436*   5439    5476    5492*   5495    5520    5528*   5531
```

```
                    5541    5552*   5555    5568    5585*   5588    5627    5635*   5638    5647    5768    5777*   5780
                    5789    5797    5817    5846    5858*   5861    5870    5881    5885    5921    5929*   5932    5941
                    5950    5968    5998    6006*   6009    6080    6089*   6092    6131    6132    6164*   6167
$TNPWR  053070     10947   10948   10968*
$TPB    001152      2044*  10785*  10796
$TPFLG  001157      2048*  10734   10796
$TPS    001150      2043*  10783   10796
$TRAP   055500      4424   11538*
$TRAP2  055522     11549*  11560
$TRP  = 000012     11553*  11562*  11563*  11564*  11565*  11566*  11568   11569*  11570*  11571*  11572*
$TRPAD  055534     11543   11560*
$TSTM   001004      2005*
$TSTNM  001102      2021*   5201*   6302*   7005    8373*   8900    11173   11208   11298   11341*  11342   11347   11351
$TYPBN= ****** U   11566
$TYPDS  053442     11101*  11565
$TYPE   052140     10734*  11382   11553   11561
$TYPEC  052352     10764   10771   10778   10783*  10784
$TYPEX  052420     10789   10791   10794*
$TYPOC  053240     11042*  11562
$TYPON  053254     11041   11044*  11564
$TYPOS  0532 4     11037*  11563
$UNIT   001332      2109*   5203*
$UNITM  001010      2007*
$USWR   001344      2116*
$VECT1  001370      2141*   4518
$VECT2  001372      2142*
$XTSTR  054410     11308   11312*
$$GET4= 000000      6318*
$.$.SM= ****** U    3356    9653    9888    10079   10090   10128   10442   10483   10495   10549   10571   10589   10616
                   10625
$OFILL  053437     11038*  11042*  11052   11087*
$40CAT= ****** U   11183   11309
.     = 064427      1959*   1963*   1968*   1970*   1972*   1980    1981*   1983*   1985*   1986*   1992    1993*   1995*
                    1997*   2018*   2098    3466*   3467*   3468*   3469*   3474*   3496*   3499*   3505*   4417    4432
                    4433    6035    6111    6115    6214    6326    6330   10796   10968   10926*  10988*  11155*  11208
                   11211   11251   11259   11276*  11350   11351   11403*
.$ASTA= ****** U   11355   11358
.$X   = 001000      1992*   1997
..HIGH  000226      1975*
..LOW   000224      1974*
```

B05


```
BMEOPS    6288#   6297
24SCOP   11293#  11307
CKEXIT    2659#   5694    5801    5821    5954    5972    8782
COMMEN    1916#
ENDCOM    1916#
ERROR     1810#   5672    6059    6068    6562    6605    6625    7460    7710    8185    8186    8188    8206    8219    8467
          8854    8992    8996    9003    9007    9011    9015    9019    9022    9025    9028    9029    9130    9135    9142
          9144    9150    9152    9161    9163    9165    9171    9183    9188    9193    9197    9201    9204    9208    9212
          9216    9220    9224    9228    9233    9238    9243    9248    9253    9258    9268    9281    9283    9293    9306
          9308    9321    9338    9347    9351    9356    9361    9366    9392    9394    9424
ESCAPE    1916#
GETPRI    1916#  11432
GETSWR    1916#
MSG       5243#   5245    5267#   5269    5291#   5293    5325#   5327    5378#   5380    5424#   5426    5476#   5478    5520#
          5522    5541#   5543    5568#   5570    5627#   5629    5769#   5770    5846#   5848    5921#   5923    5998#   6000
          6080#   6082    6130#   6134
MULT      1916#
NEWTST    1916#   5243    5267    5291    5325    5378    5424    5476    5520    5541    5568    5627    5768    5846    5921
          5998    6080    6132
NEXTAC    2658#   5646    5788    5797    5817    5869    5881    5885    5940    5950    5968
NXTAC     2657#   5647    5789    5797    5817    5870    5881    5885    5941    5950    5968
NXTST     2655#   5254    5278    5304    5352    5391    5439    5495    5531    5555    5588    5638    5780    5861    5932
          6009    6092    6167
PARMBK       1#   3240    3272
PARMDF       1#   3146
POP       1916#  10711   11142   11397   11398   11523
PUSH      1916#  10692   11101   11358   11360   11381   11503
P.DRVR       1#   3304
REGDEF       1#   2928
REPORT    1916#
R.DRIV       1#   9582
R.2DRV       1#
SCOPE     1811#   5250    5274    5300    5348    5387    5435    5491    5527    5551    5584    5634    5776    5857    5928
          6005    6088    6163    6297
SETPRI    1916#
SETTRA   11553#  11562   11563   11564   11565   11568   11569   11570   11571
SETUP     1916#   4412
SKIP      1916#
SLASH     1916#
SPACE     1916#
STARS     1759    1765    1916#   1978    1989    1991    1998    2014    2098    2101    4577    4585    4735    4749    5243
          5249    5267    5273    5291    5299    5325    5347    5378    5386    5424    5434    5476    5490    5520    5526
          5541    5550    5568    5583    5627    5633    5757    5759    5768    5775    5846    5856    5921    5927    5998
          6004    6080    6087    6132    6162    6290    6354    6356    6391    6399    6408    6415    6423    6429    6467
          6472    6502    6508    6519    6528    6548    6550    6567    6569    6589    6591    6635    6640    6673    6676
          6690    6700    6762    6772    6836    6844    6861    6866    6881    6896    6968    6972    6985    7016    7037
          7114    7122    7139    7148    7164    7173    7191    7203    7288    7302    7353    7356    7375    7380    7411
          7415    7437    7446    7468    7474    7498    7504    7517    7519    7532    7534    7538    7540    7550    7562
          7624    7633    7649    7651    7671    7680    7714    7716    7726    7729    7746    7749    7767    7771    7790
          7793    7816    7824    7861    7871    7889    7892    7902    7905    7915    7918    7929    7932    7943    7948
          7967    7970    7984    7993    8031    8040    8100    8108    8242    8252    8343    8352    8393    8398    8420
          8422    8441    8444    8497    8502    8546    8549    8571    8575    8593    8598    8629    8632    8657    8661
          8687    8697    8730    8736    8763    8779    8806    8808    8838    8844    8858    8867    8975    8985    9032
          9090    9098    9451    9455    9494    9496    9525    9527    9553    9558    9591    9622    9656    9696   10059
         10076   10094   10118   10165   10178   10197   10223   10242   10264   10322   10349   10627   10640   10683   10719
         10799   10809   10827   10838   10890   10929   10991   11014   11091   11159   11210   11216   11290   11293   11295
```

```
        11353   11410   11487   11532
SWRSU    1758#   1916#   4434#
S.DRVE      1#   3080
TEST     2656#   5243    5267    5291    5325    5378    5424    5476    5520    5541    5568    5627    5768    5846    5921
         5998    6080    6132
TRMTRP  11553#
TYPBIN   1916#
TYPDEC   1916#   6313
TYPNAM   1916#
TYPNUM   1916#
TYPOCS   1916#
TYPOCT   1916#
TYPTXT   1916#
$DECBN      1#
$OCTBN      1#  10625
$$CMRE   2012#   2051    2052    2053    2054    2055    2056    2057    2058    2059    2060    2061    2062    2063    2064
         2065    2066    2067    2068    2069    2070    2071    2072    2073    2074    2075    2076    2077    2078    2079
         2080    2081    2082
$$CMTM   2012#   2083    2084    2085    2086    2087    2088    2089    2090    2091
$$ESCA   1916#
$$NEWT   1916#   5243    5267    5291    5325    5378    5424    5476    5520    5541    5568    5627    5768    5846    5921
         5998    6080    6132
$$SET   11553#  11562   11563   11564   11565   11568   11569   11570   11571
$$SETM   4450#
$$SKIP   1916#
.EQUAT   1758#   1806
.HEADE   1758#   1768
.KT11    1758#   1916
.SETUP   1758#   4392
.SWRHI   1758#   1778
.SWRLO   1758#   1790#   1791    1792    1793    1794    1795    1796    1797
.$ACTI   1758#   1976
.$APTB   2099#
.$APTH   1758#   1987
.$APTY   1758#  11351
.$CATC   1758#   1957
.$CMTA   1758#   2012
.$DB2D   1758#  10927
.$DB2O   1758#  10888
.$EOP    1758#   6288
.$ERRO   1758#  11156
.$FCME   1758#
.$FRMD   1758#  10681
.$READC  1758#  11208
.$SAVE   1758#  11485
.$SCOP   1758#  11293
.$SIZE   1758#  11408
.$SJPR   1758#  10989
.$TRAP   1758#  11530
.$TYPD   1758#  11089
.$TYPE   1758#  10717
.$TYPO   1758#  11012

. ABS.   064427     000
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSK2:DZR6MC,DSK2:DZR6MC.SEQ SOL/CRF/NL:TOC/DOC=DRIVE9.P11 EQ:QNEWSW,DZR6MC.P11
RUN-TIME: 119.114 10 SECONDS
RUN-TIME RATIO: 889/246=3.6
CORE USED: 50K (99 PAGES)

DOCUMENT PAGES: 262

E05

Spooler runtime 43 Seconds, 179 KCS, 1172 disk reads, 0 disk writes, 262 pages

[redacted] Date 01-Dec-76 17:[redacted] Monitor IPC-0 [redacted]