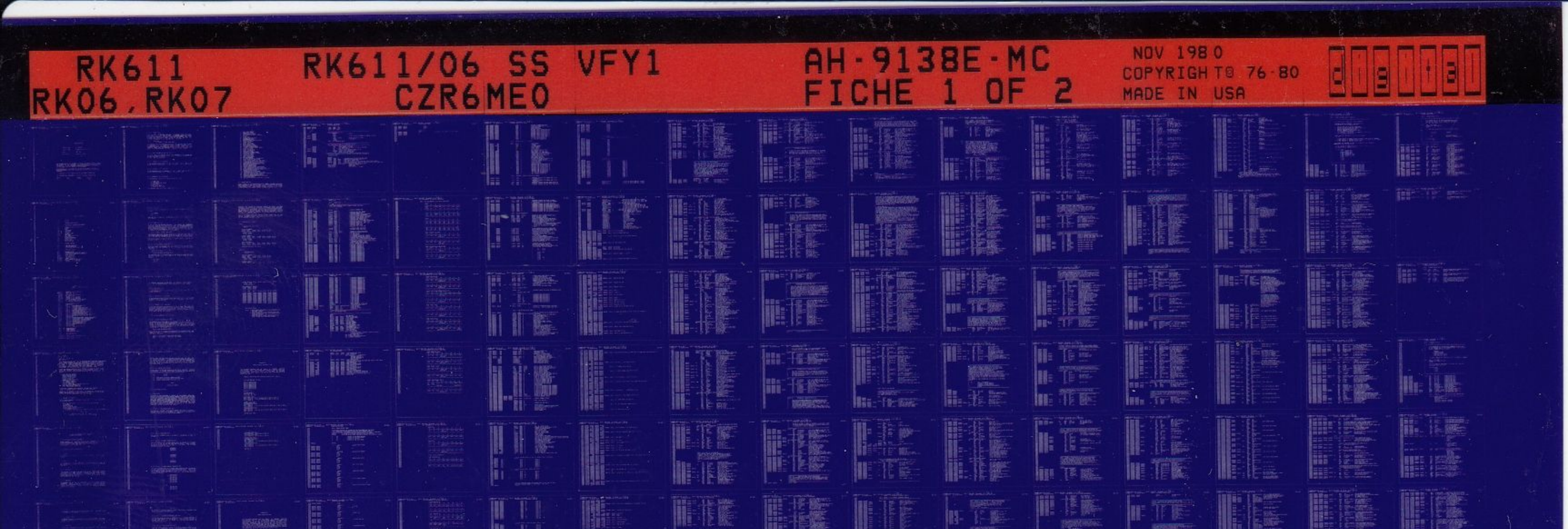
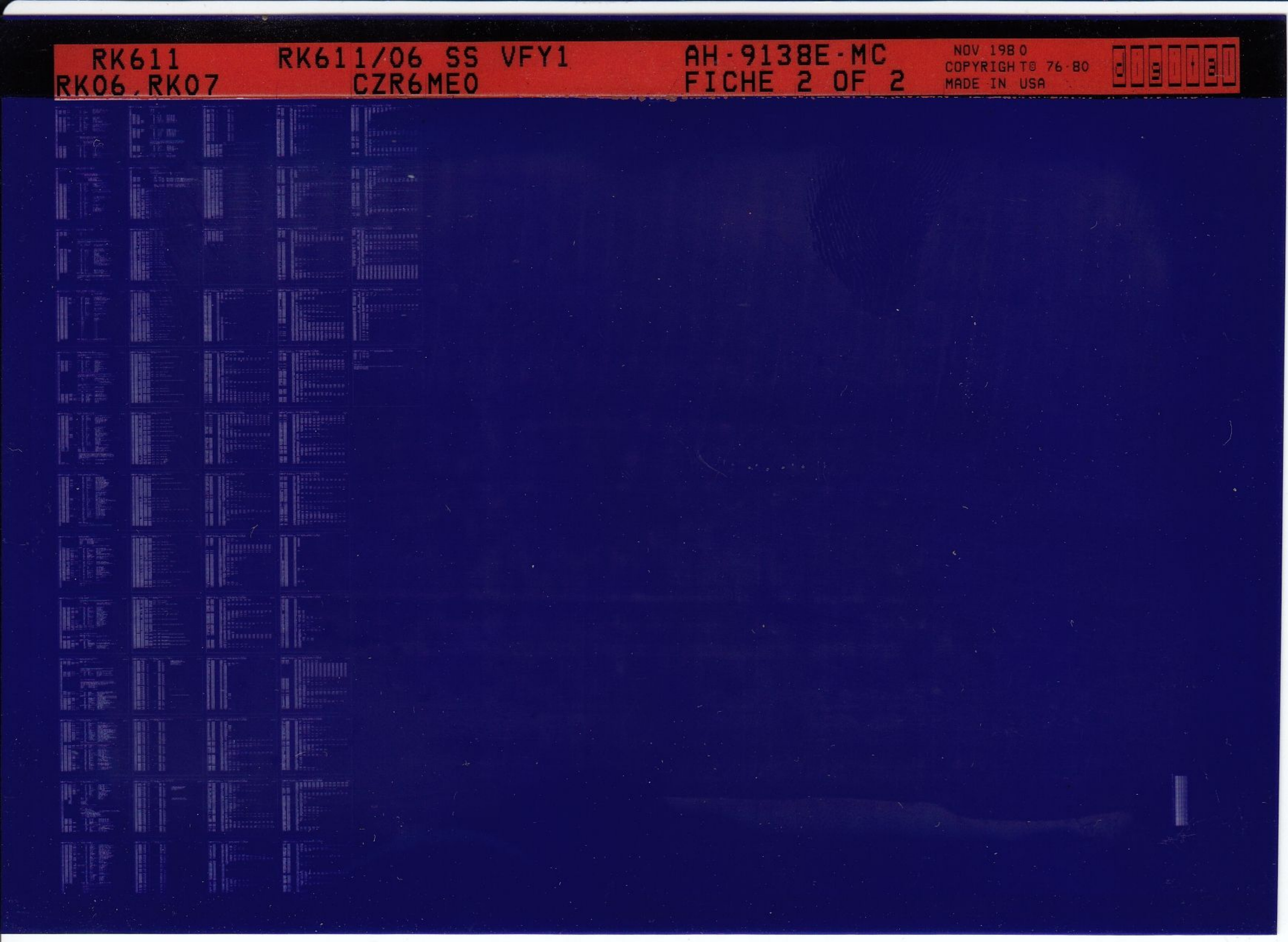
RK611
RK06,RK07

RK611/06 SS VFY1
CZR6MEO

AH-9138E-MC
FICHE 1 OF 2



RK611 RK611/06 SS VFY1 AH-9138E-MC
RK06,RK07 CZR6ME0 FICHE 2 OF 2



```
                        .REM @
```

IDENTIFICATION

PRODUCT CODE:	AC-9136E-MC
PRODUCT NAME:	CZR6ME0 RK611/06 SS VFY1
DATE:	JUNE 1980
MAINTAINER:	DIAGNOSTIC GROUP

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

1.0 ABSTRACT

THIS PROGRAM PERFORMS PART 1 OF THE RK06/RK06-RK07 SUBSYSTEM VERIFICATION TESTS, WHICH PROVIDE A FUNCTIONAL SHAKEDOWN OF THE ENTIRE SUBSYSTEM, INCLUDING THE UNIBUS INTERFACE AND ACCESS TO MAIN MEMORY. THE TESTING IN PART 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 BEGINNNG OF TESTING, THE FOLLOWING OPTIONS MAY BE SPECIFIED BY THE USER:

- RK611 REGISTER ADDRESS
- RK06/07 VECTOR ADDRESS
- RK06/07 PRIORITY LEVEL
- DRIVE(S) TO BE TESTED
- TEST(S) TO BE RUN
- NUMBER OF TEST ITERATIONS
- 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
- RK06/07 UNIBUS CONTROLLER (RK611)
- 1 TO 8 RK06/07 DRIVES
- 1 TO 8 RK06/07 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 (CZR6A THRU DZR6E AND CZR6K) AND RK06 DRIVE DIAGNOSTIC (CZR6H THRU CZR6J) SHOULD FIRST BE RUN, TO RESOLVE BASIC, SOLID HARDWARE FAULTS.

4.0 GENERAL PROGRAM CONSIDERATIONS

4.1 SYSMAC

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

4.2 XXDP

THIS PROGRAM MAY BE LOADED UNDER XXDP. SUBSYSTEM TESTS 1-21 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.
NOTE: TIMES ARE APPROX. DOUBLED FOR RK07 TESTING.

5.0 PROGRAM LOADING

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

6.0 STARTING PROCEDURE

6.1 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS OF TESTS 1-21 (PARAMETERS DEFAULTED)
204 SELECT OPERATING PARAMETERS,RK06/07 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:

```
                                                J 1
   431                                          BIT   OPTION
```

```
                                                    K 1
```

	---	------
	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: "CZR6MD RK611/RK06-RK07 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 OPERATING PARAMETERS MAY BE LISTED OR ALTERED BY VARIOUS MEANS. ALL PARAMETERS ARE TREATED AS OCTAL NUMBERS, AND THE PARAMETER LIST IS

SHARED AMONG THE TESTS.

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

FC	FIRST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, FC=0-631, DEFAULT=0
LC	LAST CYLINDER ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF CYLINDERS, LC=0-632, DEFAULT=632
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
S0	FIRST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 20-SECTOR FORMATTED CARTRIDGE. S0=0-23, DEFAULT=0
S1	LAST SECTOR ADDRESS IN TESTS WHICH ARE DONE OVER A RANGE OF SECTORS, USED ON ANY DRIVE WHICH HAS A 20-SECTOR 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,

544
545
546
547 DEFAULT=MAXIMUM BUFFER AVAILABLE, WC=0 IS INTERPRETED
548 AS 65,536 WORDS
549
550 MA PHYSICAL MEMORY ADDRESS (STARTING ADDRESS OF DATA
551 BUFFER) USED ON ALL DATA TRANSFERS IN TESTS 21, 22. MA
552 MAY BE GREATER THAN OR EQUAL TO ADDRESS OF RWBUF (NOT
553 EXCEEDING AVAILABLE MEMORY ON THE SYSTEM).
554 DEFAULT = ADDRESS OF RWBUF.
555
556 ST NUMBER OF UNIT STALL TIMES WITH WHICH TO STALL (DELAY)
557 BETWEEN RK06 COMMANDS
558
559 SM MAXIMUM NUMBER OF UNIT STALL TIMES USED IN TEST 12 ONLY
560
561
562
563 ## 8.2 SELECTION OF OPERATING PARAMETERS (ADDRESS 204 START)
564
565
566
567 ### 8.2.1 DRIVE SELECTION
568
569 AFTER AN ADDRESS 204 START, THE PROGRAM INDICATES INPUT MODE BY TYPING
570 "PARAMETER INPUT MODE". THEN, THE RK611 REGISTER ADDRESS, RK06 VECTOR
571 ADDRESS, AND RK06 PRIORITY ARE OPENED FOR POSSIBLE MODIFICATION AS
572 FOLLOWS:
573
574 RK06-07 BUS ADR = 177440 NEW=(TYPE NEW VALUE HERE)
575 RK06-07 VEC ADR = 210 NEW=(TYPE NEW VALUE HERE)
576 RK06-07 PRIORITY = 5 NEW=(TYPE NEW VALUE HERE)
577
578 NEXT THE PROGRAM TYPES THE NUMBER(S) OF THE CURRENT DRIVES UNDER TEST,
579 FOLLOWED BY A (*) ON THE NEXT LINE TO REQUEST NEW DRIVE NUMBERS. FOR
580 EXAMPLE:
581
582 PARAMETER INPUT MODE
583
584 DRIVE(S)=0,1,2,4,7
585 * (INPUT, IF ANY, GOES HERE)
586
587 THE OPERATOR TYPES THE NEW NUMBERS (SEPARATED BY COMMAS) PLUS <CR> OR
588 SIMPLY <CR> TO LEAVE THEM UNCHANGED. IF HE ENTERS DRIVE NUMBERS,
589 THESE DRIVES ARE THEN CHECKED FOR VALID STATUS BY THE PROGRAM, AND ALL
590 THOSE TYPED WHICH ARE ACCEPTABLE WILL NOW BE LISTED AS BEFORE. THE
591 OPERATOR MAY MANUALLY RECONFIGURE THE DRIVES AND TYPE IN NEW DRIVE
592 NUMBERS, AND THESE WILL BE CHECKED AND TYPED, UNTIL THE OPERATOR
593 FINALLY TYPES JUST <CR>.
594
595 ON INITIAL ENTRY FROM ADDRESS 204, THE DRIVES WHICH ARE LISTED ARE ALL
596 THOSE ON THE SUBSYSTEM WHICH ARE ON-LINE, READY, WRITE-ENABLED, AND
597 LOADED WITH A CARTRIDGE OTHER THAN AN ALIGNMENT PACK. (THIS IS THE
598 DEFAULT DRIVE SELECTION FOR ADDRESS 200 START). ON ALL OTHER ENTRIES
599 TO THE DRIVE SELECTION ROUTINE (VIA <^C> AS DESCRIBED IN SECTIONS

8.2.2.5 AND 8.2.3.6) THE PROGRAM FIRST TYPES:

TO TEST 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 R), CONTROL Z (^Z) MAY BE TYPED.

8.2.3.6 CONTROL C (^C) FUNCTION

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

8.2.4 SPECIAL PARAMETER SPECIFICATIONS

8.2.4.1 PT-DATA PATTERN SELECT WORD

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

"SELECT USER DEFINED PATTERN 15"

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

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

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

8.2.4.2 CS- CONTROL SWITCH WORD

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

BIT	OPTION
05	DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
04	TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
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
```

1C1 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, A D 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), (0-1456 FOR RK07)

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 RK06/07 DISK DRIVE SPECIFICATION.

NOTE- FOR 50 HZ OPERATION, PATCH 1 INTO LOCATION 166 (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. NOTE: SEEKS ARE TO CYL 1456 FOR THE RK07.

SAMPLE PRINTOUT --

```
     ONE CYL SEEK TIMES :
```

```
1042
1043
1044
1045                                    **FORWARD DIRECTION**
1046                                    MIN = 6332 US
1047                                    MAX = 6749 US  0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1048                                    AVG = 6536 US
1049                                    **REVERSE DIRECTION**
1050                                    MIN = 6314 US
1051                                    MAX = 6749 US  0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1052                                    AVG = 6532 US
1053
1054
1055
1056                                9.2.3  TEST 15 - AVERAGE SEEK TIME MEASUREMENT
1057
1058                                THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD  AND
1059                                REVERSE DIRECTIONS.  THE AVERAGE TIME IS CALCULATED FROM THE FOLLOWING
1060                                FORMULA :
1061                                  T AVG = [T1(410)(2)+T2(409)(2)+...+T410(1)(2)]/(410)(410)
1062                                WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS.  FORWARD AND REVERSE
1063                                TIMES  ARE  MEASURED  AND TYPED, SEPARATELY.  THE AVERAGE SEEK TIME IS
1064                                SPECIFIED TO BE < 38 MILLI-SEC.
1065
1066                                SAMPLE PRINTOUT --
1067
1068                                    AVERAGE SEEK TIMES :
1069                                    **FORWARD DIRECTION**
1070                                    AVG = 35673 US   SPEC'D MAX IS 38000 US
1071                                    **REVERSE DIRECTION**
1072                                    AVG = 35823 US   SPEC'D MAX IS 38000 US
1073
1074
1075
1076                                9.2.4  TEST 16 - MAXIMUM SEEK TIME MEASUREMENT
1077
1078                                THIS TEST MEASURES THE TIME REQUIRED TO SEEK FROM CYL  0  TO  CYL  632
1079                                OCT.   (410 DEC.).  THIS TIME REPRESENTS THE MAXIMUM SEEK TIME, AND IT
1080                                IS MEASURED 128 TIMES IN EACH DIRECTION.  THE SPECIFIED  MAXIMUM  SEEK
1081                                TIME IS < 75 MILLI-SEC.
1082
1083                                SAMPLE PRINTOUT --
1084
1085                                    MAXIMUM SEEK TIMES :
1086                                    **FORWARD DIRECTION**
1087                                    MIN = 68950 US
1088                                    MAX = 69286 US  0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1089                                    AVG = 69122 US
1090                                    **REVERSE DIRECTION**
1091                                    MIN = 70194 US
1092                                    MAX = 70667 US  0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1093                                    AVG = 70407 US
1094
1095
1096
1097                                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 RE.D 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 ;E RANGE OF DATA SECTORS AT WHICH ;RANSFERS 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 SELE';ABLE DATA PATTERNS (IN OCTAL), WITH NOTABLE FEATURES DESCRIBED:

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

	4	5	6	7
	ROTATING BOUNDARY PULSE PRECOMPENSATION	ROTATING CELL PULSE PRECOMPENSATION	ALL ZEROS	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

```
1329
1330
1331
1332                                    4.  UNIT FIELD ERROR
1333                                    5.  SUBSYSTEM TIMEOUT
1334                                    6.  SERCON PARITY ERROR
1335                                    7.  DRIVE DETECTED PARITY ERROR
1336                                    8.  AC LOW
1337                                    9.  SPEED LOSS
1338                                   10.  ILLEGAL FUNCTION ERROR
1339                                   11.  PROGRAMMING ERROR
1340                                   12.  NON-EXECUTABLE FUNCTION ERROR
1341                                   13.  DRIVE TYPE ERROR
1342                                   14.  FORMAT ERROR
1343                                   15.  WRITE LOCK ERROR
1344                                   16.  DRIVE UNSAFE ERROR
1345                                   17.  SEEK INCOMPLETE ERROR
1346                                   18.  CYLINDER OVERFLOW ERROR
1347                                   19.  ILLEGAL CYLINDER ADDRESS ERROR
1348                                   20.  DRIVE OFF TRACK
1349                                   21.  DRIVE TIMING ERROR
1350                                   22.  DATA LATE ERROR
1351                                   23.  CONTROLLER TIMEOUT ERROR
1352                                   24.  OPERATION INCOMPLETE ERROR
1353                                   25.  HEADER VRC ERROR
1354                                   26.  DATA CHECK ERROR
1355                                   27.  WRITE CHECK ERROR
1356                                   28.  DATA MISCOMPARE
1357                                   29.  NO DRIVE RESPONSE - UFE AND NXD
1358                                   30.  DRIVE ERROR WILL NOT CLEAR
1359                                   31.  DRIVE STATUS CHANGE WILL NOT CLEAR
1360                                   32.  ATTENTION BUT NO STATUS CHANGE OR FAULT
1361                                   33.  ATTENTION BUT DRIVE NOT AVAILABLE
1362                                   34.  ERROR WHILE GATHERING DRIVE STATUS
1363                                   35.  MULTIPLE DRIVE SELECT
1364                                   36.  HEADER COMPARE ERROR
1365                                   37.  ERROR IN RECALIBRATE FOR RECOVERY
1366                                   38.  CLEAR CONTROLLER DID NOT CLEAR ERROR
1367                                   39.  NO ATTENTION IN ATTENTION SUMMARY REGISTER
1368                                   40.  UNSOLICITED ATTENTION
1369                                   41.  UNEXPECTED DATA TYPE ERROR
1370                                   42.  ATTENTION DID NOT RESET WITH CLEAR
1371                                   43.  SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
1372                                   44.  DATA LATE WHEN UNLOADING HEADER
1373                                   45.  CONTROLLER ERROR WHEN DRIVER SERVICING
1374                                   46.  RETRY UNSUCCESSFUL
1375                                   47.  BAD SECTOR ERROR ON SECTOR NOT LISTED BAD
1376
1377
1378
1379                        10.2  ERROR HANDLING
1380
1381                        ERRORS REPORTED BY THE PROGRAM CONSIST OF  COMMON  FAILURES  RESULTING
1382                        FROM  ATTEMPTED  SUBSYSTEM FUNCTIONS, AS WELL AS CERTAIN ERRORS UNIQUE
1383                        TO PARTICULAR  TESTS.  EACH  ERROR  PRINTOUT  CONSISTS  OF  AN  ERROR
1384                        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
```

```
1441
1442
1443
1444                    034606  000000  000121  000000  000000  000000  036000
1445                      HI BA   LO BA
1446                      000000  074000
1447
1448                    PACK ADDRESS OF ERROR(S) :
1449                    CYLNDR  TRACK   SECTOR
1450                    000000  000000  000000
1451
1452                    WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6
1453
1454                    000000  000001  125252  000000  074000  154000  000600
1455                    000001  000002  125252  000000  074002  154002  000600
1456                    000002  000003  125252  000000  074004  154004  000600
1457                    000003  000004  125252  000000  074006  154006  000600
1458                    000004  000005  125252  000000  074010  154010  000600
1459                    000005  000006  125252  000000  074012  154012  000600
1460                    000006  000007  125252  000000  074014  154014  000600
1461                    000007  000010  125252  000000  074016  154016  000600
1462                    000010  000011  125252  000000  074020  154020  000600
1463                    000011  000012  125252  000000  074022  154022  000600
1464
1465
1466
1467
1468            11.0    REVISION HISTORY
1469
1470            .       REVE0  1) SEEK-TIMING TESTS MODIFIED IN ACCORDANCE WITH
1471                              RK07 SPEC. CHANGE.
1472                           2) CORRECTED BUG IN "INITSS" ROUTINE WHICH HAD
1473                              CAUSED INTERMITTENT "CONT. ERR. DURING DRIVER
1474                              SERVICING" ERRORS.
1475                           3) CORRECTED BUG IN "CONERR" ROUTINE WHICH HAD
1476                              CAUSED INTERMITTENT "UNDEFINED ERRORS."
```

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.

```
     CZR6M-D - RK611/RK06-RK07 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 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US
     AVG = 6518 US
     **REVERSE DIRECTION**
     MIN = 6332 US
     MAX = 6713 US     0 OF 410 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US
     AVG = 6513 US

     AVERAGE SEEK TIMES
```

```
1533
1534
1535
1536                                **FORWARD DIRECTION**
1537                                AVG = 35673 US   SPEC'D MAX IS 38000 US
1538                                **REVERSE DIRECTION**
1539                                AVG = 35823 US   SPEC'D MAX IS 38000 US
1540
1541                                MAXIMUM SEEK  IMES :
1542                                **FORWARD DIRECTION**
1543                                MIN = 68950 US
1544                                MAX = 69150 US   0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1545                                AVG = 69064 US
1546                                **REVERSE DIRECTION**
1547                                MIN = 70222 US
1548                                MAX = 70530 US   0 OF 128 ABOVE SPEC'D MAX OF 75000 US
1549                                AVG = 70379 US
1550
1551                                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.

```
CZR6M-D - RK611/RK06-RK07 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>
```

```
1608
1609
1610
1611                                    *
1612
1613                                    TEST   ITERATIONS
1614                                      1         10 * 0
1615                                      2        200 * 25
1616                                      3         10 * 0
1617                                      4         10 * 0
1618                                      5          2 * 0
1619                                      6          2 * 0
1620                                      7          2 * 4
1621                                     10        400 * 0
1622                                     11        500 * 0
1623                                     12          2 * 0
1624                                     13          1 * 0
1625                                     14          1 * 2
1626                                     15          1 * 0
1627                                     16          1 * 0
1628                                     17         10 * 0
1629                                     20         10 * 0
1630                                     21          1 *
1631
1632                                    ENTER L, C, OR I
1633                                    * L
1634
1635                                    TEST   ITERATIONS
1636                                      1           0
1637                                      2          25
1638                                      3           0
1639                                      4           0
1640                                      5           0
1641                                      6           0
1642                                      7           4
1643                                     10           0
1644                                     11           0
1645                                     12           0
1646                                     13           0
1647                                     14           2
1648                                     15           0
1649                                     16           0
1650                                     17           0
1651                                     20           0
1652                                     21           1
1653
1654                                    ENTER L, C, OR I
1655                                    * I
1656
1657                                    T = TYPE PARAMETER LIST
1658                                    O = OPEN PARAMETER LIST
1659                                    S = SET INDIVIDUAL PARAM
1660                                    R = RUN TESTS
1661
1662                                    ENTER T, O, S, OR R
1663                                    * O
```

```
1664
1665
1666
1667                                    TO DEFAULT ALL PARAMETERS TYPE D<CR>, ELSE <CR>
1668                                    *
1669                                    FC=0 *
1670                                    LC=632 * 2
1671                                    IC=1 *
1672                                    FT=0 * 1
1673                                    LT=2 * 1
1674                                    IT=1 *
1675                                    S0=0 *
1676                                    S1=23 *
1677                                    S2=0 * 10
1678                                    S3=25 * 10
1679                                    IS=1 *
1680                                    PT=0 * 20
1681                                    MA=61566 *
1682                                    WC=403 * 1100
1683                                    CS=0 *
1684                                    ST=0 *
1685                                    SM=1000 *
1686
1687                                    ENTER T, O, S, OR R
1688                                    * T
1689                                    FC=0
1690                                    LC=2
1691                                    IC=1
1692                                    FT=1
1693                                    LT=1
1694                                    IT=1
1695                                    S0=0
1696                                    S1=23
1697                                    S2=10
1698                                    S3=10
1699                                    IS=1
1700                                    PT=20
1701                                    MA=61566
1702                                    WC=1100
1703                                    CS=0
1704                                    ST=0
1705                                    SM=1000
1706
1707                                    ENTER T, O, S, OR R
1708                                    * R
1709                                    ENTER NO. OF PASSES (1-77777) :
1710                                    * 2
1711
1712                                    TESTING DRIVE 0
1713                                    DRIVE SER. NO.  8
1714                                    CART. SER. NO.  334
1715
1716                                    ONE CYL SEEK TIMES :
1717                                    **FORWARD DIRECTION**
1718                                    MIN = 6319 US
1719                                    MAX = 6691 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
```

```
1720
1721
1722
1723                                    AVG = 6502 US
1724                                    **REVERSE DIRECTION**
1725                                    MIN = 6301 US
1726                                    MAX = 6682 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1727                                    AVG = 6498 US
1728
1729                                    ONE CYL SEEK TIMES :
1730                                    **FORWARD DIRECTION**
1731                                    MIN = 6332 US
1732                                    MAX = 6695 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1733                                    AVG = 6506 US
1734                                    **REVERSE DIRECTION**
1735                                    MIN = 6314 US
1736                                    MAX = 6686 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1737                                    AVG = 6503 US
1738
1739                                    ONE CYL SEEK TIMES :
1740                                    **FORWARD DIRECTION**
1741                                    MIN = 6319 US
1742                                    MAX = 6691 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1743                                    A'G = 6504 US
1744                                    **REVERSE DIRECTION**
1745                                    MIN = 6301 US
1746                                    MAX = 6700 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1747                                    AVG = 6500 US
1748
1749                                    ONE CYL SEEK TIMES :
1750                                    **FORWARD DIRECTION**
1751                                    MIN = 6326 US
1752                                    MAX = 6698 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1753                                    AVG = 6511 US
1754                                    **REVERSE DIRECTION**
1755                                    MIN = 6317 US
1756                                    MAX = 6698 US    0 OF 410 ABOVE SPEC'D MAX OF 8000 US
1757                                    AVG = 6507 US
1758
1759                                    END PASS #    2
1760                                    TO TEST ALL DRIVES TYPE ''A''<CR>, ELSE <CR>
1761                                    *
1762
1763
1764
1765                          a
```

```
1766            000000                  PART=0
1767                                    ;*** IF "PART" IS DEFINED = 0, PART 1 WILL BE ASSEMBLED. ***
1768                                    ;*** IF "PART" IS DEFINED NOT = 0, PART 2 WILL BE ASSEMBLED. ***
1769
1770                                    ;*** REV 004 ***
1771
1772                                            .NLIST  MC,MD,CND
1773                                            .LIST   ME
1774                                            .ENABL  ABS,AMA
1775            167000                          $SWR=   167000
1776                                    ;;*****************************************************************
1777                                    .SBTTL  STARTING ADDRESSES
1778                                    ;*
1779                                    ;*      200     DEFAULT PARAMETERS FOR TESTS 1-21
1780                                    ;*      204     SELECT PARAMETERS FOR TESTS 1-21
1781                                    ;*      220     DUAL-ACCESS DATA TEST
1782                                    ;;*****************************************************************
1783
1784            000001                  $TN=1
1785                                    .TITLE  CZR6MEO RK611/06 SS VFY1
1786                                    ;*COPYRIGHT (C) 1976,1980
1787                                    ;*DIGITAL EQUIPMENT CORP.
1788                                    ;*MAYNARD, MASS. 01754
1789                                    ;*
1790                                    ;*
1791                                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1792                                    ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
1793                                    ;*
1794                                    .SBTTL  OPERATIONAL SWITCH SETTINGS
1795                                    ;*
1796                                    ;*      SWITCH                  USE
1797                                    ;*      ------          --------------------
1798                                    ;*        15            HALT ON ERROR
1799                                    ;*        14            LOOP ON TEST
1800                                    ;*        13            INHIBIT ERROR TYPEOUTS
1801                                    ;*        12            REPORT DESCRIPTION ONLY, ON ERRORS
1802                                    ;*        11            INHIBIT ITERATIONS
1803                                    ;*        10            BELL ON ERROR
1804                                    ;*         9            LOOP ON ERROR
1805                                    ;*         8            APPLY RANDOM STALL BETWEEN OPERATIONS
1806                                    ;*         7            DO EXPLICIT SEEKS IN TESTS 1-12
1807                                    ;*         6            REPORT 1 ERROR PER TRANSFER IN TESTS 17,21
1808                                    ;*         5            INHIBIT WRITES IN TEST 21
1809                                    ;*         4            INHIBIT WRITE CHECKS IN TEST 21
1810                                    ;*         3            INHIBIT READS AND SOFTWARE COMPARES IN TEST 21
1811                                    ;*         2            INHIBIT SOFTWARE COMPARES IN TEST 21
1812                                    ;*         1            READ AFTER A WRITE CHECK ERROR IN TEST 21
1813                                    ;*         0            REPORT ALL SOFTWARE COMPARE ERRORS IN TESTS 17,21
1814                                    .SBTTL  CONTROL SWITCH SETTINGS (PARAMETER CS)
1815                                    ;*
1816                                    ;*      SWITCH                  USE
1817                                    ;*      ------          --------------------
1818                                    ;*        05            DROP DRIVE IF 20(DEC) ERRORS EXCEEDED
1819                                    ;*        04            TYPE BAD SECTOR FILES (BSF'S) ON FIRST PASS
1820                                    ;*        03            INHIBIT TIMING REPORTS IN TESTS 13-16
1821
```

```
                        .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
```

```
1878            000002                  SW01=   2
1879            000001                  SW00=   1
1880                                    .EQUIV  SW09,SW9
1881                                    .EQUIV  SW08,SW8
1882                                    .EQUIV  SW07,SW7
1883                                    .EQUIV  SW06,SW6
1884                                    .EQUIV  SW05,SW5
1885                                    .EQUIV  SW04,SW4
1886                                    .EQUIV  SW03,SW3
1887                                    .EQUIV  SW02,SW2
1888                                    .EQUIV  SW01,SW1
1889                                    .EQUIV  SW00,SW0
1890
1891                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1892            100000                  BIT15=  100000
1893            040000                  BIT14=  40000
1894            020000                  BIT13=  20000
1895            010000                  BIT12=  10000
1896            004000                  BIT11=  4000
1897            002000                  BIT10=  2000
1898            001000                  BIT09=  1000
1899            000400                  BIT08=  400
1900            000200                  BIT07=  200
1901            000100                  BIT06=  100
1902            000040                  BIT05=  40
1903            000020                  BIT04=  20
1904            000010                  BIT03=  10
1905            000004                  BIT02=  4
1906            000002                  BIT01=  2
1907            000001                  BIT00=  1
1908                                    .EQUIV  BIT09,BIT9
1909                                    .EQUIV  BIT08,BIT8
1910                                    .EQUIV  BIT07,BIT7
1911                                    .EQUIV  BIT06,BIT6
1912                                    .EQUIV  BIT05,BIT5
1913                                    .EQUIV  BIT04,BIT4
1914                                    .EQUIV  BIT03,BIT3
1915                                    .EQUIV  BIT02,BIT2
1916                                    .EQUIV  BIT01,BIT1
1917                                    .EQUIV  BIT00,BIT0
1918
1919                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1920            000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
1921            000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1922            000014                  TBITVEC=14              ;;"T" BIT
1923            000014                  TRTVEC= 14              ;;TRACE TRAP
1924            000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
1925            000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1926            000024                  PWRVEC= 24              ;;POWER FAIL
1927            000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1928            000034                  TRAPVEC=34              ;;"TRAP" TRAP
1929            000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
1930            000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
1931            000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1932                                    .SBTTL  MEMORY MANAGEMENT DEFINITIONS
1933
```

```
                                ;*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  MAPL00=170200
                        170202  MAPH00=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  012450          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  012522          JMP     @#PSTART
        000220          .=220
000220  000137  012504          JMP     @#DASTRT
```

```
1990  000224  000700          ..LOW:  .WORD   700
1991  000226  001100          ..HIGH: .WORD   1100
1992                          .SBTTL  ACT11 HOOKS
1993
1994                          ;;*****************************************************************
1995                          ;HOOKS REQUIRED BY ACT11
1996          000230                  $SVPC=.                 ;SAVE PC
1997          000046                  .=46
1998  000046  025462                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1999          000052                  .=52
2000  000052  140000                  .WORD   140000          ;;2)SET LOC.52 TO 140000
2001          000230                  .=$SVPC                 ;; RESTORE PC
2002          001000                  .=1000
2003                          .SBTTL  APT PARAMETER BLOCK
2004
2005                          ;;*****************************************************************
2006                          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
2007                          ;;*****************************************************************
2008          001000                  .$X=.   ;;SAVE CURRENT LOCATION
2009          000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
2010  000024  000200                  200     ;;FOR APT START UP
2011          000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
2012  000044  001000                  $APTHDR ;;POINT TO APT HEADER BLOCK
2013          001000                  .=.$X   ;;RESET LOCATION COUNTER
2014                          ;;*****************************************************************
2015                          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
2016                          ;INTERFACE SPEC.
2017
2018  001000                  $APTHD:
2019  001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
2020  001002  001320          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
2021  001004  001320          $TSTM:  .WORD   1320    ;;RUN TIM OF LONGEST TEST
2022  001006  001546          $PASTM: .WORD   1546    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
2023  001010  001546          $UNITM: .WORD   1546    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDIT.ONAL UNIT
2024  001012  000030                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
2025          120210          AVECT1=120210           ;RKVEC=210, RKPRI=5
2026          177440          ABASE=177440            ;RKBAS ADRS
2027          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          $TCNT:  .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)
```

```
2084  001224  000000                $REG21: .WORD   0               ;;CONTAINS (($REGAD)+42)
2085  001226  000000                $REG22: .WORD   0               ;;CONTAINS (($REGAD)+44)
2086  001230  000000                $REG23: .WORD   0               ;;CONTAINS (($REGAD)+46)
2087  001232  000000                $REG24: .WORD   0               ;;CONTAINS (($REGAD)+50)
2088  001234  000000                $REG25: .WORD   0               ;;CONTAINS (($REGAD)+52)
2089  001236  000000                $REG26: .WORD   0               ;;CONTAINS (($REGAD)+54)
2090  001240  000000                $REG27: .WORD   0               ;;CONTAINS (($REGAD)+56)
2091  001242  000000                $REG30: .WORD   0               ;;CONTAINS (($REGAD)+60)
2092  001244  000000                $REG31: .WORD   0               ;;CONTAINS (($REGAD)+62)
2093  001246  000000                $PEG32: .WORD   0               ;;CONTAINS (($REGAD)+64)
2094  001250  000000                $REG33: .WORD   0               ;;CONTAINS (($REGAD)+66)
2095  001252  000000                $REG34: .WORD   0               ;;CONTAINS (($REGAD)+70)
2096  001254  000000                $REG35: .WORD   0               ;;CONTAINS (($REGAD)+72)
2097  001256  000000                $REG36: .WORD   0               ;;CONTAINS (($REGAD)+74)
2098  001260  000000                $REG37: .WORD   0               ;;CONTAINS (($REGAD)+76)
2099  001262  000000                $TMP0:  .WORD   0               ;;USER DEFINED
2100  001264  000000                $TMP1:  .WORD   0               ;;USER DEFINED
2101  001266  000000                $TMP2:  .WORD   0               ;;USER DEFINED
2102  001270  000000                $TMP3:  .WORD   0               ;;USER DEFINED
2103  001272  000000                $TMP4:  .WORD   0               ;;USER DEFINED
2104  001274  000000                $TMP5:  .WORD   0               ;;USER DEFINED
2105  001276  000000                $TMP6:  .WORD   0               ;;USER DEFINED
2106  001300  000000                $TMP7:  .WORD   0               ;;USER DEFINED
2107  001302  000000                $TMP10: .WORD   0               ;;USER DEFINED
2108  001304  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
2109  001306  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
2110  001310  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
2111  001314     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
2112  001315     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
2113  001316  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
2114                                ;;*****************************************************************
2115                                .SBTTL  APT MAILBOX-ETABLE
2116
2117                                ;;*****************************************************************
2118                                .EVEN
2119  001320                        $MAIL:                          ;;APT MAILBOX
2120  001320  000000                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
2121  001322  000000                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
2122  001324  000000                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
2123  001326  000000                $PASS:  .WORD   APASS           ;;PASS COUNT
2124  001330  000000                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
2125  001332  000000                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
2126  001334  000000                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
2127  001336  000000                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
2128  001340                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
2129  001340     000                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
2130  001341     000                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
2131  001342  000000                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
2132  001344  000000                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
2133  001346  000000                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
2134                                ;*                              BITS 15-11=CPU TYPE
2135                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
2136                                ;*                                      11/70=06,PDQ=07,Q=10
2137                                ;*                              BIT 10=REAL TIME CLOCK
2138                                ;*                              BIT  9=FLOATING POINT PROCESSOR
2139                                ;*                              BIT  8=MEMORY MANAGEMENT
```

```
2140  001350  000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
2141  001351  000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
2142                              ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
2143                              ;*                              900 NSEC CORE=001
2144                              ;*                              300 NSEC BIPOLAR=002
2145                              ;*                              500 NSEC MOS=003
2146  001352  000000              $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
2147                              ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
2148  001354  000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
2149  001355  000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
2150  001356  000000              $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
2151  001360  000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
2152  001361  000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
2153  001362  000000              $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
2154  001364  000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
2155  001365  000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
2156  001366  000000              $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
2157  001370  120210              $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
2158  001372  000000              $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
2159  001374  177440              $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
2160  001376  000377              $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
2161  001400                      $ETEND:
2162                              .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  056325                         EM1
001402  060751                         DH100
001404  062450                         DT100
001406  062564                         DF01

                                        ;ERROR 2                ;NON-EXISTANT MEMORY
001410  056344                         EM2
001412  060751                         DH100
001414  062450                         DT100
001416  062564                         DF01

                                        ;ERROR 3                ;NON-EXISTANT DRIVE
001420  056362                         EM3
001422  060751                         DH100
001424  062450                         DT100
001426  062564                         DF01

                                        ;ERROR 4                ;UNIT FIELD ERROR
001430  056400                         EM4
001432  060751                         DH100
001434  062450                         DT100
001436  062564                         DF01

                                        ;ERROR 5                ;SUBSYSTEM TIMEOUT
001440  056417                         EM5
001442  060751                         DH100
001444  062450                         DT100
001446  062614                         DF02

                                        ;ERROR 6
001450  056436                         EM6                     ;D TO C PARITY ERROR
001452  060751                         DH100
001454  062450                         DT100
001456  062650                         DF03

                                        ;ERROR 7
001460  056455                         EM7                     ;DRIVE DETECTED PARITY ERROR
001462  060751                         DH100
001464  062450                         DT100
```

```
2219  001466  062650                        DF03
2220
2221                                        ;ERROR 10
2222  001470  056474                        EM10                ;AC LOW
2223  001472  060751                        DH100
2224  001474  062450                        DT100
2225  001476  062614                        DF02
2226
2227                                        ;ERROR 11
2228  001500  056503                        EM11                ;SPEED LOSS
2229  001502  060751                        DH100
2230  001504  062450                        DT100
2231  001506  062614                        DF02
2232
2233                                        ;ERROR 12
2234  001510  056516                        EM12                ;ILLEGAL FUNCTION
2235  001512  060751                        DH100
2236  001514  062450                        DT100
2237  001516  062614                        DF02
2238
2239                                        ;ERROR 13
2240  001520  056530                        EM13                ;PROGRAMMING ERROR
2241  001522  060751                        DH100
2242  001524  062450                        DT100
2243  001526  062564                        DF01
2244
2245                                        ;ERROR 14           ;NON-EXISTANT FUNCTION
2246  001530  056540                        EM14
2247  001532  060751                        DH100
2248  001534  062450                        DT100
2249  001536  062614                        DF02
2250
2251                                        ;ERROR 15
2252  001540  056560                        EM15                ;DRIVE TYPE ERROR
2253  001542  060751                        DH100
2254  001544  062450                        DT100
2255  001546  062614                        DF02
2256
2257                                        ;ERROR 16
2258  001550  056574                        EM16                ;FORMAT ERROR
2259  001552  060751                        DH100
2260  001554  062450                        DT100
2261  001556  062614                        DF02
2262
2263                                        ;ERROR 17
2264  001560  056604                        EM17                ;WRITE LOCK ERROR
2265  001562  060751                        DH100
2266  001564  062450                        DT100
2267  001566  062614                        DF02
2268
2269                                        ;ERROR 20
2270  001570  056621                        EM20                ;DRIVE UNSAFE
2271  001572  060751                        DH100
2272  001574  062450                        DT100
2273  001576  062614                        DF02
2274
```

```
2275                                    ;ERROR 21
2276  001600  056634                   EM21                    ;SEEK INCOMPLETE
2277  001602  060751                   DH100
2278  001604  062450                   DT100
2279  001606  062614                   DF02
2280
2281                                    ;ERROR 22
2282  001610  056645                   EM22                    ;CYLINDER OVERFLOW
2283  001612  060751                   DH100
2284  001614  062450                   DT100
2285  001616  062614                   DF02
2286
2287                                    ;ERROR 23
2288  001620  056660                   EM23                    ;ILLEGAL CYLINDER
2289  001622  060751                   DH100
2290  001624  062450                   DT100
2291  001626  062614                   DF02
2292
2293                                    ;ERROR 24
2294  001630  056675                   EM24                    ;DRIVE OFF TRACK
2295  001632  060751                   DH100
2296  001634  062450                   DT100
2297  001636  062614                   DF02
2298
2299                                    ;ERROR 25
2300  001640  056711                   EM25                    ;DRIVE TIMING ERROR
2301  001642  060751                   DH100
2302  001644  062450                   DT100
2303  001646  062614                   DF02
2304
2305                                    ;ERROR 26
2306  001650  056730                   EM26                    ;DATA LATE
2307  001652  060751                   DH100
2308  001654  062450                   DT100
2309  001656  062614                   DF02
2310
2311                                    ;ERROR 27
2312  001660  056742                   EM27                    ;CONTROLLER TIMEOUT
2313  001662  060751                   DH100
2314  001664  062450                   DT100
2315  001666  062614                   DF02
2316
2317                                    ;ERROR 30               ;OPERATION INCOMPLETE
2318  001670  056757                   EM30
2319  001672  060751                   DH100
2320  001674  062450                   DT100
2321  001676  062724                   DF05
2322
2323                                    ;ERROR 31
2324  001700  056771                   EM31                    ;HEADER VRC ERROR
2325  001702  060751                   DH100
2326  001704  062450                   DT100
2327  001706  062724                   DF05
2328
2329                                    ;ERROR 32
2330  001710  057005                   EM32                    ;DATA CHECK ERROR
```

```
2331  001712  060751                  DH100
2332  001714  062450                  DT100
2333  001716  062760                  DF07
2334
2335                                  ;ERROR 33
2336  001720  057022                  EM33                ;WRITE CHECK ERROR
2337  001722  060751                  DH100
2338  001724  062450                  DT100
2339  001726  063014                  DF10
2340
2341                                  ;ERROR 34
2342  001730  057036                  EM34                ;DATA MISCOMPARE(S)
2343  001732  060751                  DH100
2344  001734  062450                  DT100
2345  001736  062710                  DF04
2346
2347                                  ;ERROR 35
2348  001740  057056                  EM35                ;NO DRIVE RESPONSE-UFE & NXD
2349  001742  060751                  DH100
2350  001744  062450                  DT100
2351  001746  062564                  DF01
2352
2353                                  ;ERROR 36
2354  001750  057110                  EM36                ;DRIVE ERROR WILL NOT CLEAR
2355  001752  000000                  0
2356  001754  000000                  0
2357  001756  000000                  0
2358
2359                                  ;ERROR 37           ;DRIVE STATUS CHANGE WILL NOT CLEAR
2360  001760  057137                  EM37
2361  001762  000000                  0
2362  001764  000000                  0
2363  001766  000000                  0
2364
2365                                  ;ERROR 40
2366  001770  057200                  EM40                ;ATTENTION BUT NO STATUS CHANGE OR FAULT
2367  001772  060751                  DH100
2368  001774  062450                  DT100
2369  001776  062614                  DF02
2370
2371                                  ;ERROR 41
2372  002000  057244                  EM41                ;ATTENTION BUT DRIVE NOT AVAILABLE
2373  002002  060751                  DH100
2374  002004  062450                  DT100
2375  002006  062614                  DF02
2376
2377                                  ;ERROR 42
2378  002010  057274                  EM42                ;ATTENTION WHEN NOT EXPECTED
2379  002012  060751                  DH100
2380  002014  062450                  DT100
2381  002016  062614                  DF02
2382
2383                                  ;ERROR 43           ;ERROR WHILE GATHERING DRIVE STATUS
2384  002020  057324                  EM43
2385  002022  060751                  DH100
2386  002024  062450                  DT100
```

```
2387  002026  063060                    DF12
2388
2389                                    ;ERROR 44
2390  002030  057524                    EM63            ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2391  002032  060751                    DH100
2392  002034  062450                    DT100
2393  002036  063060                    DF12
2394
2395                                    ;ERROR 45
2396  002040  057562                    EM64            ;NO ATTENTION IN ATTENTION SUMMARY REG
2397  002042  060751                    DH100
2398  002044  062450                    DT100
2399  002046  063060                    DF12
2400
2401                                    ;ERROR 46
2402  002050  057610                    EM65            ;UNSOLICITED ATTENTION
2403  002052  060751                    DH100
2404  002054  062450                    DT100
2405  002056  063060                    DF12
2406
2407                                    ;ERROR 47
2408  002060  057632                    EM66            ;UNEXPECTED DATA TYPE ERROR
2409  002062  060751                    DH100
2410  002064  062450                    DT100
2411  002066  063060                    DF12
2412
2413                                    ;ERROR 50
2414  002070  057651                    EM67            ;ATTENTION DID NOT RESET WITH CLEAR
2415  002072  060751                    DH100
2416  002074  062450                    DT100
2417  002076  063060                    DF12
2418
2419                                    ;ERROR 51
2420  002100  057710                    EM70            ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE ATTENTION
2421  002102  060751                    DH100
2422  002104  062450                    DT100
2423  002106  063060                    DF12
2424
2425                                    ;ERROR 52
2426  002110  057355                    EM52            ;MULTIPLE DRIVE SELECT
2427  002112  060751                    DH100
2428  002114  062450                    DT100
2429  002116  063060                    DF12
2430
2431                                    ;ERROR 53
2432  002120  057372                    EM53            ;ABREVIATED HCE ERROR
2433  002122  060751                    DH100
2434  002124  062450                    DT100
2435  002126  063110                    DF13
2436
2437                                    ;ERROR 54
2438  002130  056757                    EM30            ;OPERATION INCOMPLETE ERROR
2439  002132  060751                    DH100
2440  002134  062450                    DT100
2441  002136  063140                    DF14
2442
```

```
2443                                    ;ERROR 55
2444  002140  056771                    EM31                ;ABREVIATED HVRC ERROR
2445  002142  060751                    DH100
2446  002144  062450                    DT100
2447  002146  063110                    DF13
2448
2449                                    ;ERROR 56
2450  002150  057407                    EM56                ;2 TIMEOUT ERROR
2451  002152  060751                    DH100
2452  002154  062450                    DT100
2453  002156  063170                    DF15
2454
2455                                    ;ERROR 57           ;2ND LEVEL IN SUBSYSTEM TIMEOUT
2456  002160  057407                    EM56
2457  002162  060751                    DH100
2458  002164  062450                    DT100
2459  002166  063234                    DF16
2460
2461                                    ;ERROR 60
2462  002170  057426                    EM60                ;ERROR IN RECAL FOR RECOVERY
2463  002172  000000                    0
2464  002174  000000                    0
2465  002176  000000                    0
2466
2467                                    ;ERROR 61
2468  002200  057460                    EM61                ;ABORT MESSAGE
2469  002202  000000                    0
2470  002204  000000                    0
2471  002206  000000                    0
2472
2473                                    ;ERROR 62
2474  002210  057510                    EM62                ;CYLINDER MISCOMPARE
2475  002212  060751                    DH100
2476  002214  062450                    DT100
2477  002216  063300                    DF17
2478
2479                                    ;ERROR 63           ;DATA ERROR WORDS
2480  002220  000000                    0
2481  002222  000000                    0
2482  002224  062542                    DT602
2483  002226  063410                    DF25
2484
2485                                    ;ERROR 64
2486  002230  057524                    EM63                ;CLEAR CONTROLLER DID NOT CLEAR ERROR
2487  002232  060751                    DH100
2488  002234  062450                    DT100
2489  002236  062614                    DF02
2490
2491                                    ;ERROR 65
2492  002240  057562                    EM64                ;NO ATTEN"ION IN ATTENTION SUMMARY REG
2493  002242  060751                    DH100
2494  002244  062450                    DT100
2495  002246  062614                    DF02
2496
2497                                    ;ERROR 66
2498  002250  057610                    EM65                ;UNSOLICITED ATTENTION
```

```
2499  002252  060751                  DH100
2500  002254  062450                  DT100
2501  002256  062614                  DF02
2502
2503                                  ;ERROR 67
2504  002260  057632                  EM66          ;UNEXPECTED DATA TYPE ERROR
2505  002262  060751                  DH100
2506  002264  062450                  DT100
2507  002266  062614                  DF02
2508
2509                                  ;ERROR 70
2510  002270  057651                  EM67          ;ATTENTION DID NOT RESET WITH CLEAR
2511  002272  060751                  DH100
2512  002274  062450                  DT100
2513  002276  062614                  DF02
2514
2515                                  ;ERROR 71
2516  002300  057710                  EM70          ;SUBSYSTEM CLEAR DID NOT CLEAR ATT
2517  002302  060751                  DH100
2518  002304  062450                  DT100
2519  002306  062614                  DF02
2520
2521                                  ;ERROR 72
2522  002310  057755                  EM71          ;DATA LATE WHEN UNLOADING HEADER
2523  002312  060751                  DH100
2524  002314  062450                  DT100
2525  002316  062614                  DF02
2526
2527                                  ;ERROR 73
2528  002320  060010                  EM72          ;CONTROLLER ERROR DURING DRIVER SERVICE
2529  002322  060751                  DH100
2530  002324  062450                  DT100
2531  002326  062614                  DF02
2532
2533                                  ;ERROR 74
2534  002330  060043                  EM73          ;DRIVE DETECTED PARITY ERROR
2535  002332  060751                  DH100
2536  002334  062450                  DT100
2537  002336  062614                  DF02
2538
2539                                  ;ERROR 75
2540  002340  060063                  EM74          ;UNDEFINED ERROR
2541  002342  060751                  DH100
2542  002344  062450                  DT100
2543  002346  062614                  DF02
2544
2545                                  ;ERROR 76
2546  002350  060075                  EM75          ;MARKING SECTOR BAD MESSAGE
2547  002352  000000                  0
2548  002354  000000                  0
2549  002356  000000                  0
2550
2551                                  ;ERROR 77
2552  002360  060125                  EM76          ;BAD DATA VERIFICATION WITH READ
2553  002362  061663                  DH605
2554  002364  062534                  DT601
```

```
2555  002366  063334                    DF21
2556
2557                                    ;ERROR 100
2558  002370  060207                    EM77                ;RETRY SUCCESFUL MESSAGE
2559  002372  000000                    0
2560  002374  062534                    DT601
2561  002376  063374                    DF23
2562
2563                                    ;ERROR 101
2564  002400  060207                    EM77                ;ANOTHER RETRY SUCCESSFUL MESSAGE
2565  002402  000000                    0
2566  002404  062534                    DT601
2567  002406  063374                    DF23
2568
2569                                    ;ERROR 102
2570  002410  060230                    EM100               ;RETRY UNSUCCESSFUL MESSAGE
2571  002412  000000                    0
2572  002414  062534                    DT601
2573  002416  063374                    DF23
2574
2575                                    ;ERROR 103
2576  002420  060253                    EM101               ;NO VALID HEADERS IN TRACK JUST READ
2577  002422  061645                    DH6042
2578  002424  062534                    DT601
2579  002426  063404                    DF24
2580
2581                                    ;ERROR 104
2582  002430  060331                    EM102               ;BSE ERROR ON SECTOR NOT LISTED AS BAD
2583  002432  060751                    DH100
2584  002434  062450                    DT100
2585  002436  062614                    DF02
2586
2587                                    ;ERROR  105
2588  002440  060375                    EM103               ;TIMED-OUT ON READ HEADER
2589  002442  060751                    DH100
2590  002444  062450                    DT100
2591  002446  062650                    DF03
2592
2593                                    ;ERROR  106
2594  002450  060423                    EM104               ;TIMED-OUT ON SEEK
2595  002452  060751                    DH100
2596  002454  062450                    DT100
2597  002456  062650                    DF03
2598
2599                                    ;ERROR  107
2600  002460  060445                    EM105               ;DRIVE SIEZED BY OTHER PORT
2601  002462  060751                    DH100
2602  002464  062450                    DT100
2603  002466  062614                    DF02
2604
2605                                    ;ERROR  110
2606  002470  060476                    EM106               ;''DATA MISCMPR WHILE BAI SET''
2607  002472  060751                    DH100
2608  002474  062450                    DT100
2609  002476  063424                    DF27
2610
```

```
2611                                    ;ERROR  111
2612  002500  060531                   EM107                   ;'NO NEM WHEN EXPECTED''
2613  002502  000000                   0
2614  002504  000000                   0
2615  002506  000000                   0
2616
2617                                    ;ERROR  112
2618  002510  060574                   EM110                   ;''INTRPT WHEN CNTRLR NOT READY''
2619  002512  060751                   DH100
2620  002514  062450                   DT100
2621  002516  062564                   DF01
2622
2623                                    ;ERROR  113
2624  002520  060627                   EM111                   ;'NO ATT'N ON SEEK''
2625  002522  060751                   DH100
2626  002524  062450                   DT100
2627  002526  062614                   DF02
2628
2629                                    ;ERROR  114
2630  002530  060650                   EM112                   ;DRIVE'S CYLINDER INCORRECT
2631  002532  062072                   DH702
2632  002534  062510                   DT202
2633  002536  063414                   DF26
2634
2635                                    ;ERROR  115
2636  002540  000000                   0                       ;TYPE ADRS OF DATA MISCOMPARE(S)
2637  002542  000000                   0
2638  002544  062534                   D1601
2639  002546  063040                   DF11
2640
2641                                    ;ERROR  116
2642  002550  057036                   EM34                    ;DATA MISCOMPARE (11/70)
2643  002552  061227                   DH103
2644  002554  062560                   DT103
2645  002556  063324                   DF20
2646
2647                                    ;ERROR  117
2648  002560  000000                   0                       ;PART OF DATA MISCOMPARE
2649  002562  000000                   0
2650  002564  062450                   DT100
2651  002566  063344                   DF22
2652
2653                                    ;ERROR  120
2654  002570  060672                   EM113                   ;ABORT- CAN'T READ BSF
2655  002572  060751                   DH100
2656  002574  062450                   DT100
2657  002576  062564                   DF01
2658
2659                                    ;ERROR  121
2660  002600  060720                   EM114                   ;KT11 FAILURE
2661  002602  060751                   DH100
2662  002604  062450                   DT100
2663  002606  063450                   DF30
2664
2665                                    ;ERROR  122
2666  002610  060735                   EM115                   ;MEM PARITY ERROR
```

```
2667  002612  060751                          DH100
2668  002614  062450                          DT100
2669  002616  063474                          DF31
2670
2671                                  ;ERROR  123
2672  002620  056362                 EM3                     ;NED WHEN SIZING UNDER APT DRIVE TABLE
2673  002622  000000                 0
2674  002624  000000                 0
2675  002626  000000                 0
2676
2677
2678
2679
2680
2681
```

```
                        .SBTTL  BIT ASSIGNMENTS IN THE RK611 REGISTERS

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

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

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

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

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

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

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

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

```
2736                    ;RKDB (DATA BUFFER)
2737                    ; 15     14     13     12     11     10     9      8
2738                    ;-----------------------------------------------------
2739                    ;DB15 ! DB14 ! DB13 ! DB12 ! DB11 ! DB10 ! DB9  ! DB8  !
2740                    ;                    READ/WRITE                       !
2741                    ;-----------------------------------------------------
2742
2743                    ;  7      6      5      4      3      2      1      0
2744                    ;-----------------------------------------------------
2745                    ; DB7 ! DB6  ! DB5  ! DB4  ! DB3  ! DB2  ! DB1  ! DB0  !
2746                    ;                    READ/WRITE                       !
2747                    ;-----------------------------------------------------
2748
2749                    ;RKASOF (ATTENTION SUMMARY AND OFFSET)
2750                    ; 15     14     13     12     11     10     9      8
2751                    ;-----------------------------------------------------
2752                    ;ATN7 ! ATN6 ! ATN5 ! ATN4 ! ATN3 ! ATN2 ! ATN1 ! ATN0 !
2753                    ;                    READ ONLY                        .
2754                    ;-----------------------------------------------------
2755
2756                    ;  7      6      5      4      3      2      1      0
2757                    ;-----------------------------------------------------
2758                    ; UN  ! OF6  ! OF5  ! OF4  ! OF3  ! OF2  ! OF1  ! OF0  !
2759                    ;  ?  !              READ/WRITE                       !
2760                    ;-----------------------------------------------------
2761
2762                    ;RKWC (WORD COUNT)
2763                    ; 15     14     13     12     11     10     9      8
2764                    ;-----------------------------------------------------
2765                    ;WC15 ! WC14 ! WC13 ! WC12 ! WC11 ! WC10 ! WC9  ! WC8  !
2766                    ;                    READ/WRITE                       !
2767                    ;-----------------------------------------------------
2768
2769                    ;  7      6      5      4      3      2      1      0
2770                    ;-----------------------------------------------------
2771                    ; WC7 ! WC6  ! WC5  ! WC4  ! WC3  ! WC2  ! WC1  ! WC0  !
2772                    ;                    READ/WRITE                       !
2773                    ;-----------------------------------------------------
2774
2775                    ;RKBA (BUS ADDRESS)
2776                    ; 15     14     13     12     11     10     9      8
2777                    ;-----------------------------------------------------
2778                    ;BA15 ! BA14 ! BA13 ! BA12 ! BA11 ! BA10 ! BA9  ! BA8  !
2779                    ;                    READ/WRITE                       !
2780                    ;-----------------------------------------------------
2781
2782                    ;  7      6      5      4      3      2      1      0
2783                    ;-----------------------------------------------------
2784                    ; BA7 ! BA6  ! BA5  ! BA4  ! BA3  ! BA2  ! BA1  ! BA0  !
2785                    ;                    READ/WRITE                !ALWYS0.
2786                    ;-----------------------------------------------------
```

```
                        ;RKER (ERROR REGISTER)
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ; DCK ! UNS  ! OPI  ! DTE  ! WLE  ! IDAE ! COE  ! HVRC !
                        ;                    READ ONLY                        !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ; BSE ! ECH  ! DTYPE! FMTE ! DRPAR! ILF  ! SKI  ! ILC  !
                        ;                    READ ONLY                        !
                        ;-----------------------------------------------------
                        ;
                        ;RKDS (DRIVE STATUS)
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ;SVAL ! DSC  ! PIP  ! SPON ! WRL  !  UN  !  UN  ! DTP  !
                        ;                    READ ONLY                        !
                        ;-----------------------------------------------------
                        ;
                        ;  7      6      5      4      3      2      1      0
                        ;-----------------------------------------------------
                        ;DRDY .  VV  ! DROT ! SPLS ! ACLO ! OFSET!  UN  ! DRA  !
                        ;                    READ ONLY                        !
                        ;-----------------------------------------------------
                        ;
                        ;RKMR1 (MAINTENANCE REG 1)
                        ; 15     14     13     12     11     10     9      8
                        ;-----------------------------------------------------
                        ;RD   ! WRT  ! ECCW ! PCD  ! PCA  ! MEWD ! MERD ! 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                       !
                        ;-----------------------------------------------------
```

```
2826                                    ;RKECC/PAT
2827                                    ; 15     14     13     12     11     10     9      8
2828                                    ;-----------------------------------------------------
2829                                    ; UN  !  UN  !  UN  !  UN  !  UN  ! EPA10! EPA9 ! EPA8 !
2830                                    ;                    READ ONLY                        !
2831                                    ;-----------------------------------------------------
2832
2833                                    ;  7      6      5      4      3      2      1      0
2834                                    ;-----------------------------------------------------
2835                                    ;EPA7 ! EPA6 ! EPA5 ! EPA4 ! EPA3 ! EPA2 ! EPA1 ! EPA0 !
2836                                    ;                    READ ONLY                        !
2837                                    ;-----------------------------------------------------
2838
2839                                    ;RKECC/POS
2840                                    ; 15     14     13     12     11     10     9      8
2841                                    ;-----------------------------------------------------
2842                                    ; UN  !  UN  !  UN  ! EPO12! EPO11! EPO10! EPO9 . EPO8 '
2843                                    ;                    READ ONLY
2844                                    ;-----------------------------------------------------
2845
2846                                    ;  7      6      5      4      3      2      1      0
2847                                    ;-----------------------------------------------------
2848                                    ;EPO7 ! EPO6 ! EPO5 ! EPO4 ! EPO3 ! EPO2 ! EPO1 . EPO0 .
2849                                    ;                    READ ONLY                        .
2850                                    ;-----------------------------------------------------
2851
2852                                    ;DRIVE STATUS INFORMATION
2853
2854                                    ;LINE A MESSAGE 00
2855                                    ; 15     14     13     12     11     10     9      8
2856                                    ;-----------------------------------------------------
2857                                    ; PAR ! DSC  ! PIP  ! SPON ! WRLK ! OPON! FMT  ! DRTP !
2858                                    ;-----------------------------------------------------
2859
2860                                    ;  7      6      5      4      3      2      1      0
2861                                    ;-----------------------------------------------------
2862                                    ;DRDY !  VV    DRA  .  NU     NU  ! DRIVE SELECT CODE  !
2863                                    ;-----------------------------------------------------
```

```
2864                                    ;LINE B MESSAGE 00
2865                                    ; 15     14     13     12     11     10     9      8
2866                                    ;-----------------------------------------------------
2867                                    ; PAR ! UNS  ! DROT ! DCLO ! WLE  ! SKI  ! PERR ! ILF  .
2868                                    ;-----------------------------------------------------
2869
2870                                    ;  7     6      5      4      3      2      1      0
2871                                    ;-----------------------------------------------------
2872                                    ; FLT ! ACLO ! IVDA !  UN  .  UN  .  UN  !  0  !  0  '
2873                                    ;-----------------------------------------------------
2874
2875                                    ;LINE A MESSAGE 01
2876                                    ; 15     14     13     12     11     10     9      8
2877                                    ;-----------------------------------------------------
2878                                    ; PAR ' UNLDG! RTZ  . LDG  . REV  ! FWD  ! SPEED! CART !
2879                                    ;     ! HDS  .     ! HDS  !      !      ! OK   ! PRES !
2880                                    ;-----------------------------------------------------
2881
2882                                    ;  7     6      5      4      3      2      1      0
2883                                    ;-----------------------------------------------------
2884                                    ;DOOR ! BRUSH! HEADS! TRK  !  UN  !  DRIVE SELECT CODE !
2885                                    ;LTCH ! HOME ! HOME ! FOLOW!      !                     !
2886                                    ;     !      !      ! OK   .      .                     !
2887                                    ;-----------------------------------------------------
2888
2889                                    ;LINE B MESSAGE 01
2890                                    ; 15     14     13     12     11     10     9      8
2891                                    ;-----------------------------------------------------
2892                                    ; PAR ! SERVO! LIMIT! SEEK ! PLO  ! DiBIT! INDEX! MULTI!
2893                                    ;     ! BRAKE! ON SK!NO MO ! ERR  !  ERR ! ERR  !HD SEL!
2894                                    ;-----------------------------------------------------
2895
2896                                    ;  7     6      5      4      3      2      1      0
2897                                    ;-----------------------------------------------------
2898                                    ;HEAD !WR GTE!WR CUR! SEC  !  NU  !  NU  !  0  !  1  '
2899                                    ;FAULT!AND NO!AND NO! ERR  .      !      !     !     !
2900                                    ;     !XISTON!WR GTE!      .      !      !     !     !
2901                                    ;-----------------------------------------------------
```

```
                        ;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
```

```
2958    000012          RKDS=   12              ;DRIVE STATUS REGISTER
2959    000014          RKER=   14              ;ERROR REGISTER
2960    000016          RKASOF= 16              ;ATTENTION SUMMARY AND OFFSET REGISTER
2961    000020          RKDC=   20              ;DESIRED CYLINDER REGISTER
2962    000020          RKDCYL= 20              ;DESIRED CYLINDER REGISTER
2963    000024          RKDB=   24              ;DATA BUFFER
2964    000026          RKMR1=  26              ;MAINTENANCE REGISTER 1
2965    000034          RKMR2=  34              ;MAINTENANCE REGISTER 2
2966    000036          RKMR3=  36              ;MAINTENANCE REGISTER 3
2967    000030          RKPOS=  30              ;ECC POSITION INFORMATION
2968    000030          RKECPS= 30              ;ECC POSITION INFORMATION
2969    000032          RKPAT=  32              ;ECC PATTERN INFORMATION
2970    000032          RKECPT= 32              ;ECC PATTERN INFORMATION
2971
2972                    .SBTTL  DRIVE COMMANDS
2973
2974    000101          SELDRV= 101             ;SELECT DRIVE
2975    000103          PACK=   103             ;PACK ACKNOWLEDGE
2976    000105          CLEAR=  105             ;DRIVE CLEAR
2977    000107          UNLOAD= 107             ;UNLOAD
2978    000111          SRTSPL= 111             ;START SPINDLE
2979    000113          RECAL=  113             ;RECALIBRATE
2980    000115          OFFSET= 115             ;OFFSET
2981    000117          SEEK=   117             ;SEEK
2982    000121          RDDATA= 121             ;READ DATA
2983    000123          WRDATA= 123             ;WRITE DATA
2984    000125          RDHEAD= 125             ;READ HEADER
2985    000127          WRHEAD= 127             ;WRITE HEADER AND DATA
2986    000131          WRTCHK= 131             ;WRITE CHECK
2987
2988                    ;       THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER
2989                    ;       TO SIMULATE A SPECIFIC DESIRED OPERATION
2990
2991    000140          RELEAS= 140             ;RELEASE DRIVE
2992    000141          RDSTAT= 141             ;GET ALL STATUS FROM DRIVE
2993    000164          RDALHD= 164             ;READ ALL HEADERS
2994    000176          CONCLR= 176             ;CONTROLLER CLEAR (BIT 15 OF CS1)
2995    000177          SUBCLR= 177             ;SUBSYSTEM CLEAR (BIT 5 OF CS2)
2996    000300          INTR=   300             ;GENERATE INTERRUPT TO CPU
2997
2998                    ;       DRIVER ISSUED SERVICE COMMANDS
2999
3000    000001          DR.SEL= 001             ;DRIVE SELECT
3001    000005          DR.CLR= 005             ;DRIVE CLEAR
3002
3003                    .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
3004
3005    000001          GO=     BIT0            ;GO BIT
3006    000100          IE=     BIT6            ;INTERRUPT ENABLE
3007    000200          RDY=    BIT7            ;CONTROLLER READY
3008    000400          BA16=   BIT8            ;BUS ADDRESS BIT 16
3009    001000          BA17=   BIT9            ;BUS ADDRESS BIT 17
3010    002000          CDT=    BIT10           ;CONTROLLER DRIVE TYPE (0-RK06,1=RK07)
3011    004000          CTO=    BIT11           ;CONTROLLER TIMED OUT WAITING FOR
3012                                            ;  DRIVE RESPONSE
3013    010000          CFMT=   BIT12           ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1-20 SECTOR)
```

```
3014            020000                  SPAR=   BIT13                    ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
3015            040000                  DI=     BIT14                    ;DRIVE INTERRUPT
3016            100000                  CERR=   BIT15                    ;CONTROLLER ERROR
3017            100000                  CCLR=   BIT15                    ;CONTROLLER CLEAR
3018
3019                                    ;       THESE BIT DEFINITIONS ARE USED FOR ADDRESS
3020                                    ;       THE HIGH BYTE OF RKCS1
3021
3022            000001                  B.BA16= BIT0                     ;BUS ADDRESS BIT 16
3023            000002                  B.BA17= BIT1                     ;BUS ADDRESS BIT 17
3024            000004                  B.CDT=  BIT2                     ;CONTROLLER DRIVE TYPE (0=RK06,1=RK07)
3025            000020                  B.CFMT= BIT4                     ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=20 SECTOR)
3026
3027                                    .SBTTL  CONTROL AND STATUS REGISTER 2 BITS
3028
3029            000007                  DRVMSK= 7                        ;MASK FOR DRIVE SELECTION CODE
3030            000010                  DESL=   BIT3                     ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3031            000010                  RLS=    BIT3                     ;DESELECT OR RELEASE DRIVE IN BITS 0-2
3032            000020                  BAI=    BIT4                     ;BUS ADDRESS INCREMENT INHIBIT
3033            000040                  CLR=    BIT5                     ;CLEAR CONTROLLER AND ALL DRIVES
3034            000040                  SCLR=   BIT5                     ;CLEAR CONTROLLER AND ALL DRIVES
3035            000100                  IR=     BIT6                     ;INPUT READY
3036            000200                  OR=     BIT7                     ;OUTPUT READY
3037            000400                  UFE=    BIT8                     ;UNIT FIELD ERROR
3038            001000                  MDS=    BIT9                     ;MULTIPLE DRIVE SELECT
3039            002000                  PGE=    BIT10                    ;PROGRAMMING ERROR
3040            004000                  NEM=    BIT11                    ;NON-EXISTENT MEMORY
3041            010000                  NED=    BIT12                    ;NON-EXISTENT DRIVE
3042            020000                  UPE=    BIT13                    ;UNIBUS PARITY ERROR
3043            040000                  WCE=    BIT14                    ;WRITE CHECK ERROR
3044            100000                  DLT=    BIT15                    ;DATA LATE ERROR
3045
3046                                    .SBTTL  ERROR REGISTER BIT DEFINITION
3047
3048            000001                  ILC=    BIT0                     ;ILLEGAL FUNCTION CODE
3049                                    ;*ILF=  BIT0                     ;ILLEGAL FUNCTION CODE
3050            000002                  SKI=    BIT1                     ;SEEK INCOMPLETE
3051            000004                  ILF=    BIT2                     ;ILLEGAL DRIVE FUNCTION
3052            000004                  NXF=    BIT2                     ;ILLEGAL DRIVE FUNCTION
3053            000010                  DRPAR=  BIT3                     ;DRIVE DETECTED DRIVE BUS PARITY ERROR
3054            000020                  FMTE=   BIT4                     ;FORMAT ERROR
3055            000040                  DTYE=   BIT5                     ;DRIVE TYPE ERROR
3056            000100                  ECH=    BIT6                     ;ECC HARD
3057            000200                  BSE=    BIT7                     ;BAD SECTOR ERROR
3058            000400                  HCRC=   BIT8                     ;HEADER CRC ERRROR
3059            000C400                 HVRC=   BIT8                     ;HEADER VRC ERRROR
3060            001000                  COE=    BIT9                     ;CYLINDER ADDRESS OVERFLOW ERROR
3061            002000                  IDAE=   BIT10                    ;INVALID DISK ADDRESS ERROR
3062            004000                  WLE=    BIT11                    ;WRITE LOCK ERROR
3063            010000                  DTE=    BIT12                    ;DRIVE TIMING ERROR
3064            020000                  OPI=    BIT13                    ;OPERATION (SEARCH) INCOMPLETE
3065            040000                  UNS=    BIT14                    ;DRIVE UNSAFE
3066            100000                  DCK=    BIT15                    ;DATA CHECK
3067
3068                                    .SBTTL  STATUS REGISTER BIT DEFINITION
3069
```

```
3070            000001                  DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
3071                                                                    ;  THIS BIT IS RESET)
3072            000004                  OFST=   BIT2                    ;DRIVE OFFSET
3073            000010                  ACLO=   BIT3                    ;AC LOW
3074            000020                  SPDLSS= BIT4                    ;SPEED LOSS
3075            000020                  DCLO=   BIT4                    ;DC LOW
3076            000040                  DROT=   BIT5                    ;DRIVE OFF TRACK
3077            000100                  VV=     BIT6                    ;VOLUME VALID
3078            000200                  DRY=    BIT7                    ;DRIVE READY
3079            000200                  DRDY=   BIT7                    ;DRIVE READY
3080            000400                  DDT=    BIT8                    ;DRIVE TYPE (0=RK06,1=RK07)
3081            004000                  WRL=    BIT11                   ;WRITE LOCK
3082            020000                  PIP=    BIT13                   ;POSITIONING IN PROGRESS
3083            040000                  DSC=    BIT14                   ;DRIVE STATUS CHANGE
3084            100000                  SVAL=   BIT15                   ;STATUS VALID
3085
3086                                    .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
3087
3088            000017                  MESMSK= 17                      ;MESSAGE MASK
3089
3090            000020                  PAT=    BIT4                    ;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
3091            000040                  DMD=    BIT5                    ;DIAGNOSTIC MODE
3092            000100                  MSP=    BIT6                    ;MAINTENANCE SECTOR PULSE
3093            000200                  MIND=   BIT7                    ;MAINTENANCE INDEX
3094            000400                  MCLK=   BIT8                    ;MAINTENANCE CLOCK
3095            001000                  MERD=   BIT9                    ;MAINTENANCE ENCODED READ DATA
3096            002000                  MEWD=   BIT10                   ;MAINTENACNE ENCODED WRITE DATA
3097            004000                  PCA=    BIT11                   ;PRECOMPENSATION ADVANCE
3098            010000                  PCD=    BIT12                   ;PRECOMPENSATION DELAY
3099            020000                  ECCW=   BIT13                   ;ECC WORD IS BEING READ OR WRITTEN
3100            040000                  WRTGAT= BIT14                   ;WRITE GATE
3101            100000                  RDGATE= BIT15                   ;READ GATE
3102
3103                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
3104
3105            000040                  S.DRA=  BIT5                    ;DRIVE AVAILIABLE
3106            000100                  S.VV=   BIT6                    ;VOLUME VALID
3107            000200                  S.DRY=  BIT7                    ;DRIVE READY
3108            000400                  S.TYPE= BIT8                    ;DRIVE TYPE
3109            001000                  S.FORM= BIT9                    ;DRIVE FORMAT
3110            002000                  S.OFF=  BIT10                   ;OFFSET
3111            004000                  S.WRL=  BIT11                   ;WRITE LOCK
3112            010000                  S.SPIN= BIT12                   ;SPINDLE ON
3113            020000                  S.PIP=  BIT13                   ;POSITIONING IN PROGRESS
3114            040000                  S.DSC=  BIT14                   ;DRIVE STATUS CHANGE
3115
3116                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
3117
3118            000040                  S.ICYL= BIT5                    ;ILLEGAL CYLINDER ADDRESS
3119            000100                  S.ACLO= BIT6                    ;AC LOW
3120            000200                  S.FLT=  BIT7                    ;DRIVE FAULT
3121            000400                  S.ILF=  BIT8                    ;ILLEGAL FUNCTION
3122            001000                  S.PAR=  BIT9                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
3123            002000                  S.SKI=  BIT10                   ;SEEK INCOMPLETE
3124            004000                  S.WLE=  BIT11                   ;WRITE LOCK ERROR
3125            010000                  S.SPLS= BIT12                   ;SPEED LOSS
```

```
3126            010000                  S.DCLO= BIT12                   ;DC LOW
3127            020000                  S.DROT= BIT13                   ;DRIVE OFF TRACK
3128            040000                  S.UNS=  BIT14                   ;DRIVE UNSAFE
3129
3130                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
3131
3132            000020                  S.XDOK= BIT4                    ;TRANSDUCER OK
3133            000040                  S.HDHM= BIT5                    ;HEADS HOME
3134            000100                  S.BRHM= BIT6                    ;BRUSHES HOME
3135            000200                  S.DOOR= BIT7                    ;DOOR INTERLOCKED
3136            000400                  S.CART= BIT8                    ;CARTRAGE INTERLOCK
3137            001000                  S SPOK= BIT9                    ;SPEED OK
3138            002000                  S.FWD=  BIT10                   ;FORWARD
3139            004000                  S.REV=  BIT11                   ;REVERSE
3140            010000                  S.LOAD= BIT12                   ;HEADS LOADING
3141            020000                  S.RTZ=  BIT13                   ;RETURN TO ZERO
3142            040000                  S.UNLD= BIT14                   ;HEADS UNLOADING
3143
3144                                    .SBTTL  DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
3145
3146            000020                  S.SECT= BIT4                    ;SECTOR ERROR
3147            000040                  S.WCLK= BIT5                    ;WRITE CLOCK AND NO WRITE GATE
3148            000100                  S.WGAT= BIT6                    ;WRITE GATE AND NO TRANSISTIONS
3149            000200                  S.HDFL= BIT7                    ;HEAD FAULT
3150            000400                  S.MHD=  BIT8                    ;MULTIPLE HEAD SELECT
3151            001000                  S.XERR= BIT9                    ;INDEX ERROR
3152            002000                  S.DIB=  BIT10                   ;DIBIT ERROR
3153            004000                  S.PLO=  BIT11                   ;PLO ERROR
3154            010000                  S.NMOV= BIT12                   ;SEEK AND NO MOTION
3155            020000                  S.LIMD= BIT13                   ;LIMIT DETECT ON SEEK
3156            040000                  S.BRKE= BIT14                   ;SERVO-BRAKE
3157
3158                                    .SBTTL  COMMON MASKS
3159
3160            000007                  M.DRV=  7                       ;DRIVE CODE
3161            100000                  M.PAR=  BIT15                   ;PARITY
3162            000003                  M.ID=   3                       ;BYTE ID
3163            017760                  M.CDIF= 17760                   ;CYLINDER DIFFERENCE/OFFSET
3164            017760                  M.CADD= 17760                   ;CYLINDER ADDRESS
3165            077770                  M.SER=  77770                   ;DRIVE SERIAL NUMBER
3166            000760                  M.SECT= 760                     ;SECTOR COUNT
3167            007000                  M.HEAD= 7000                    ;HEAD DECODE
```

```
                        .SBTTL  PARAMETER BLOCK ALLOCATION

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

                        .SBTTL  PARAMETERS PASSED TO THE DRIVER

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

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

                        .SBTTL  PROGRAM DEVICE STATUS REGISTER DEFINITION

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

```
3224            000100                  CMDTO=  BIT6                    ;NO TERMINATION TO COMMAND FOR AT
3225                                                                    ;  LEAST 1 SECOND
3226            000200                  W.WCK=  BIT7                    ;WRITE FOR WRITE WRITE CHECK
3227            000400                  NOCHK=  BIT8                    ;NO CHECK, DO NOT SET INTERRUPT ENABLE
3228            001000                  PBSVAL= BIT9                    ;PARAMETER STATUS WORDS VALID
3229                                                                    ;  (SET WHEN ERROR TERMINATION OR
3230                                                                    ;  READ STATUS COMMAND)
3231            002000                  DRPDRV= BIT10                   ;DROP DRIVE FROM TEST SEQUENCE
3232            004000                  NODSC=  BIT11                   ;ATTENTION SET BUT DCS AND FAULT RESET
3233            010000                  DRVSZD= BIT12                   ;DRIVE SEIZED BY OTHER PORT
3234            020000                  E.UNLD= BIT13                   ;DRIVE UNLOADED DUE TO ERROR
3235            040000                  C.INIT= BIT14                   ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
3236            100000                  DTBAII= BIT15                   ;INHIBIT BUS ADDRESS INCREMENT
3237
3238                                    .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
3239
3240                                    ;       THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
3241                                    ;         FROM THE DRIVER TO THE CALLING PROGRAM
3242
3243            000016                  P.CS1=  16                      ;COMMAND AND STATUS REGISTER 1
3244            000020                  P.CS2=  20                      ;COMMAND AND STATUS REGISTER 2
3245            000022                  P.WCR=  22                      ;WORD COUNT REGISTER
3246            000024                  P.BAR=  24                      ;BUS ADDRESS REGISTER
3247            000026                  P.DTS=  26                      ;DESIRED TRACK SECTOR REGISTER
3248            000030                  P.DCYL= 30                      ;DESIRED CYLINDER REGISTER
3249            000032                  P.ASOF= 32                      ;ATTENTION SUMMARY/OFFSET REGISTER
3250            000034                  P.ER=   34                      ;ERROR REGISTER
3251            000036                  P.DS=   36                      ;STATUS REGISTER
3252            000040                  P.A00=  40                      ;MESSAGE A STATUS BYTE 00
3253            000042                  P.B00=  42                      ;MESSAGE B STATUS BYTE 00
3254            000044                  P.A01=  44                      ;MESSAGE A STATUS BYTE 01
3255            000046                  P.B01=  46                      ;MESSAGE B STATUS BYTE 01
3256            000050                  P.A10=  50                      ;MESSGGE A STATUS BYTE 10
3257            000052                  P.B10=  52                      ;MESSAGE B STATUS BYTE 10
3258            000054                  P.A11=  54                      ;MESSAGE A STATUS BYTE 11
3259            000056                  P.B11=  56                      ;MESSAGE B STATUS BYTE 11
3260            000060                  P.EPOS= 60                      ;ECC POSITION INFORMATION
3261            000062                  P.EPAT= 62                      ;ECC PATTERN INFORMATION
3262
3263                                    .SBTTL  PARAMETER BLOCK 0 FOR DRIVE
3264
3265   002630       000                 PARM0:  .BYTE   0               ;DRIVE NUMBER
3266   002631       000                         .BYTE   0               ;COMMAND
3267   002632    000000                         .WORD   0               ;CYLINDER ADDRESS
3268   002634       000                         .BYTE   0               ;SECTOR ADDRESS
3269   002635       000                         .BYTE   0               ;TRACK ADDRESS
3270   002636       000                         .BYTE   0               ;OFFSET VALUE
3271   002637       000                         .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3272   002640    000000                         .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3273   002642    000000                         .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3274   002644    000000                         .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
3275   002646    000000                         .WORD   0               ;COMMAND AND STATUS REGISTER 1
3276   002650    000000                         .WORD   0               ;COMMAND AND STATUS REGISTER 2
3277   002652    000000                         .WORD   0               ;WORD COUNT REGISTER
3278   002654    000000                         .WORD   0               ;BUS ADDRESS REGISTER
3279   002656    000000                         .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
```

```
3280  002660  000000                    .WORD   0               ;DESIRED CYLINDER REGISTER
3281  002662  000000                    .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3282  002664  000000                    .WORD   0               ;ERROR REGISTER
3283  002666  000000                    .WORD   0               ;STATUS REGISTER
3284  002670  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3285  002672  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3286  002674  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3287  002676  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3288  002700  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3289  002702  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3290  002704  000000                    .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3291  002706  000000                    .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3292  002710  000000                    .WORD   0               ;ECC POSITION INFORMATION
3293  002712  000000                    .WORD   0               ;ECC PATTERN INFORMATION
3294
3295                          .SBTTL  PARAMETER BLOCK 1 FOR DRIVE
3296
3297  002714     000          PARM1:  .BYTE   0               ;DRIVE NUMBER
3298  002715     000                  .BYTE   0               ;COMMAND
3299  002716  000000                  .WORD   0               ;CYLINDER ADDRESS
3300  002720     000                  .BYTE   0               ;SECTOR ADDRESS
3301  002721     000                  .BYTE   0               ;TRACK ADDRESS
3302  002722     000                  .BYTE   0               ;OFFSET VALUE
3303  002723     000                  .BYTE   0               ;BUS ADDRESS (BITS 16 AND 17)
3304  002724  000000                  .WORD   0               ;BUS ADDRESS (BITS 0 - 15)
3305  002726  000000                  .WORD   0               ;WORD COUNT (2'S COMPLEMENT)
3306  002730  000000                  .WORD   0               ;PROGRAM DRIVE STATUS INFORMATION
3307  002732  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 1
3308  002734  000000                  .WORD   0               ;COMMAND AND STATUS REGISTER 2
3309  002736  000000                  .WORD   0               ;WORD COUNT REGISTER
3310  002740  000000                  .WORD   0               ;BUS ADDRESS REGISTER
3311  002742  000000                  .WORD   0               ;DESIRED TRACK AND SECTOR REGISTER
3312  002744  000000                  .WORD   0               ;DESIRED CYLINDER REGISTER
3313  002746  000000                  .WORD   0               ;ATTENTION SUMMARY/OFFSET REGISTER
3314  002750  000000                  .WORD   0               ;ERROR REGISTER
3315  002752  000000                  .WORD   0               ;STATUS REGISTER
3316  002754  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 00
3317  002756  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 00
3318  002760  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 01
3319  002762  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 01
3320  002764  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 10
3321  002766  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 10
3322  002770  000000                  .WORD   0               ;MESSAGE LINE A STATUS BYTE 11
3323  002772  000000                  .WORD   0               ;MESSAGE LINE B STATUS BYTE 11
3324  002774  000000                  .WORD   0               ;ECC POSITION INFORMATION
3325  002776  000000                  .WORD   0               ;ECC PATTERN INFORMATION
3326
3327                          .SBTTL  TEMPORARY CONTROLLER REGISTER STORAGE
3328
3329  003000  000000          T.CS1:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3330                                                          ;  REGISTER 1
3331  003002  000000          T.CS2:  .WORD   0               ;TEMPORARY STORAGE FOR COMMAND AND STATUS
3332                                                          ;  REGISTER 2
3333  003004  000000          T.WCR:  .WORD   0               ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
3334  003006  000000          T.BA:   .WORD   0               ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
3335  003010  000000          T.DA:   .WORD   0               ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
```

```
3336  003012  000000              T.DC:   .WORD   0               ;TEMPORARY STORAGE FOR DRIVE CYLINDER
3337  003014  000000              T.ASOF: .WORD   0               ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
3338                                                              ;  AND OFFSET
3339  003016  000000              T.ER:   .WORD   0               ;TEMPORARY STORAGE FOR ERROR REGISTER
3340  003020  000000              T.DS:   .WORD   0               ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
3341  003022  000000              T.MR1:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
3342  003024  000000              T.MR2:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
3343  003026  000000              T.MR3:  .WORD   0               ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
3344  003030  000000              T.POS:  .WORD   0               ;TEMPORARY STORAGE FOR ECC POSITION
3345  003032  000000              T.PAT:  .WORD   0               ;TEMPORARY STORAGE FOR ECC PATTERN
3346  003034  000000              T.DB:   .WORD   0               ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER
3347
3348                              .SBTTL  DRIVER PARAMERTERS
3349
3350  003036  177440              RKBAS:  .WORD   177440          ;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
3351  003040  000210              RKVEC:  .WORD   210             ;ADDRESS OF R611 VECTOR
3352  003042  000240              RKPRI:  .WORD   PR5             ;RK611 INTERRUPT PRIORITY
3353  003044  040652              A.NORM: ERRFRE                  ;ADDRESS OF NORMAL RETURN FROM DRIVER
3354  003046  042130              A.ABNL: ERRHDL                  ;ADDRESS OF ABNORMAL RETURN FROM DRIVER
3355  003050  041414              A.CONT: CONERR                  ;ADDRESS OF CONTROLLER ERROR RETURN
3356  003052  000000              E.CONT: .WORD   0               ;CONTROLLER ERROR STATUS
3357                                                              ;  THIS LOCATION IS CLEARED WHEN EVERY COMMAND
3358                                                              ;  IS INITIATED.  IF A CONTROLLER ERROR
3359                                                              ;  OCCURS THE FOLLOWING BIT ASSIGNMENT IS
3360                                                              ;  USED:
3361                                                              ;
3362          000001              E.CCLR= BIT0                    ;CLEAR CONTROLLER DID NOT CLEAR ERROR
3363          000002              E.NOAT= BIT1                    ;NO ATTENTION IN ATTENTION SUMMARY REG
3364          000004              E.UATT= BIT2                    ;UNSOLICATED ATTENTION (SEQUENTIAL ONLY)
3365          000010              E.UDAT= BIT3                    ;UNEXPECTED DATA TYPE ERROR
3366          000020              E.CLAT= BIT4                    ;ATTENTION DID NOT RESET WITH CLEAR
3367          000040              E.SCLR= BIT5                    ;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
3368                                                              ;  ATTENTION
3369          000100              E.ILLD= BIT6                    ;ILLEGAL DRIVER COMMAND
3370          000400              E.DLT=  BIT8                    ;DATA LATE WHEN UNLOADING HEADER
3371          001000              E.CERR= BIT9                    ;CONTROLLER ERROR DURING DRIVER SERVICING
3372          002000              E.DPAR= BIT10                   ;DRIVE DETECTED PARITY ERROR
3373          040000              E.CMTO= BIT14                   ;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
3374          100000              E.MDS=  BIT15                   ;MULTIPLE DRIVE SELECT
3375
3376  003054  000000              O.WAIT: .WORD   0               ;PARAMETER BLOCK OF THE DRIVE
3377                                                              ;WAITING FOR COMMAND COMPLETION
3378  003056  000400              W.MTIM: .WORD   400             ;LOOP COUNTER FOR MILISECOND SCAN OF DRIVE
3379  003060  000400              W.MILI: .WORD   400             ;16 MILISECOND TIME FOR PROGRAM
3380                                                              ;
3381                                                              ;       CPU     VALUE
3382                                                              ;       ---     -----
3383                                                              ;
3384                                                              ;       11/05   100
3385                                                              ;       11/10
3386                                                              ;       11/20
3387                                                              ;       11/34
3388                                                              ;       11/40
3389                                                              ;       11/45   400
3390                                                              ;       11/50
3391                                                              ;       11/70
```

```
3392  003062  000300                  W.SEC:  .WORD   300             ;SECOND COUNT COUNT FOR ALL COMMANDS
3393                                                                  ;  EXCEPT START SPINDLE
3394  003064  003000                  W.8SEC: .WORD   3000            ;8 SECOND FOR DRIVE CYCLE DOWN
3395  003066  030000                  W.MIN:  .WORD   30000           ;MINUTE TIME FOR START SPINDLE
3396  003070  000000                  HDR.AD: .WORD   0               ;ADDRESS USED FOR READ ALL HEADERS
3397  003072  000000                  HDR.CT: .WORD   0               ;NUMBER OF HEADERS LEFT TO READ FOR READ
3398                                                                  ;  ALL HEADERS
3399  003074     000                  I.ISRL: .BYTE   0               ;INTERRUPT OR RELEASED COMMAND ISSUED
3400  003075     002      004     010 H.HEAD: .BYTE   2,4,10          ;HEAD DECODES
3401  003100     000                  W.TIME: .BYTE   0               ;DRIVES BEING WATCH-DOG TIMED
3402
3403                                  .SBTTL  INTERRUPT MASKS
3404
3405  003101     000                  INTMSK: .BYTE   0               ;INTERRUPT MASKS FOR DRIVE IN PARAMETER BLOCK
3406
3407                                  ;       INTERRUPT MASK TABLE
3408
3409  003102     001                  I.DRV:  .BYTE   1               ;INTERRUPT MASK FOR DRIVE 0
3410  003103     002                          .BYTE   2               ;INTERRUPT MASK FOR DRIVE 1
3411  003104     004                          .BYTE   4               ;INTERRUPT MASK FOR DRIVE 2
3412  003105     010                          .BYTE   10              ;INTERRUPT MASK FOR DRIVE 3
3413  003106     020                          .BYTE   20              ;INTERRUPT MASK FOR DRIVE 4
3414  003107     040                          .BYTE   40              ;INTERRUPT MASK FOR DRIVE 5
3415  003110     100                          .BYTE   100             ;INTERRUPT MASK FOR DRIVE 6
3416  003111     200                          .BYTE   200             ;INTERRUPT MASK FOR DRIVE 7
3417
3418                                  .SBTTL  PARAMETER BLOCK TABLE
3419
3420  003112  002630                  PBLKT:  PARM0                   ;ADDRESS OF PARAMETER BLOCK GIVEN WITH
3421                                                                  ;DRIVE CALL.  MUST BE LOADED INTO PBLKT
3422
3423
3424                                  .SBTTL  TIME FOR WATCH-DOG TIMER
3425
3426  003114  000000                  W.DRV:  .WORD   0               ;TIME FOR INSTRUCTION IN PARAMETER BLOCK
3427                                  .SBTTL  PROGRAM SPECIFIC RESERVED LOCATIONS
3428  003116     000                  MDFLAG: .BYTE   0               ;FLAG TO INDIC. DEFLT OR PARAM MODE
3429  003117     000                  XXDPCH: .BYTE   0               ;XXDP CHAIN MODE FLAG
3430  003120     000                  TSTING: .BYTE   0               ;CURRENTLY RUNNING TESTS IF = 1
3431  003121     000                  DERCNT: .BYTE   0               ;DATA ERROR COUNT
3432  003122     000                  OPCOMP: .BYTE   0               ;OPERATION COMPLETE FLAG
3433  003123     000                  DONE:   .BYTE   0               ;DONE SWITCH
3434  003124     000                  TYPFMT: .BYTE   0               ;DRIVE TYPE & FORMAT CONTROL
3435  003125     000                  FORMAT: .BYTE   0               ;DRIVE FORMAT IN BIT 4 OF BYTE
3436  003126     000                  ERRCNT: .BYTE   0               ;ERROR COUNT
3437  003127     004                  ERRLMT: .BYTE   4               ;ERROR LIMIT
3438  003130     000                  DRVERS: .BYTE   0               ;ERROR COUNT FOR CURRENT DRIVE
3439  003131     000                  OPCONT: .BYTE   0               ;OPERATION CONTROL SWITCHES
3440  003132     000                  PCLKF:  .BYTE   0               ;IF BYTE=1, KW11-P CLOCK IS PRESENT
3441  003133     000                  DOTIM:  .BYTE   0               ;IF BYTE=1, DO TIMING TESTS
3442  003134     000                  XOVLAD: .BYTE   0               ;FLAG = 1 IF XXDP IS POSSIBLY OVERLAID
3443  003135     000                  XDPSVD: .BYTE   0               ;FLAG = 1 IF XXDP IS SAVED
3444  003136     000                  DULACS: .BYTE   0               ;FLAG=1 IF DUAL ACCESS TEST
3445  003137     000                  DRNAFG: .BYTE   0               ;=1 INDICATES DRIVE SIEZED BY OTHER PORT
3446  003140     000                  REISSU: .BYTE   0               ;DUAL-ACC FLAG TO RE-ISSUE COMMAND
3447  003141     000                  WCEFLG: .BYTE   0               ;WRITE CHECK ERROR FLAG
```

```
3448  003142  000                 DLTFLG: .BYTE   0               ;DATA LATE ERROR FLAG
3449  003143  000                 NORTRY: .BYTE   0               ;"NO-RETRY" FLAG
3450  003144  000                 UBMPRS: .BYTE   0               ;UNIBUS MAP PRESENT IF = 1
3451  003145  000                 MEMABT: .BYTE   0               ;SET BYTE = 1 FOR NO ABORT
3452                                                              ;    ON MEMORY PARITY ERRORS
3453  003146  000                 HLPOVL: .BYTE   0               ;HELP FILE OVERLAID INDICATOR
3454  003147  000                 NOTYPE: .BYTE   0               ;INDICATES PROGRAM JUST LOADED IF 0
3455          000001              WHDSW=BIT0                      ;WRITE HEADER & DATA SWITCH
3456          000002              VFHDSW=BIT1                     ;VERIFY HEADERS SWITCH
3457          000004              WCDASW=BIT2                     ;WRITE CHECK DATA SWITCH
3458          000010              RCDASW=BIT3                     ;READ CHECK DATA SWITCH
3459          000020              CREQSW=BIT4                     ;OFFSET REQUIRED SWITCH
3460                                      .EVEN
3461
3462  003150  177546              LKS:    .WORD   177546          ;KW11-L CLOCK STATUS REGISTER
3463  003152  172540              PKS:    .WORD   172540          ;KW11-P CONTROL AND STATUS REGISTER
3464  003154  172542              PKSB:   .WORD   172542          ;KW11-P COUNT SET BUFFER REGISTER
3465  003156  172544              PKRB:   .WORD   172544          ;KW11-P COUNTER REGISTER
3466  003160  000100              LCVEC:  .WORD   100             ;KW11-L VECTOR STORAGE
3467  003162  000104              PCVEC:  .WORD   104             ;KW11-P VECTOR STORAGE
3468          000114              MEMVEC=114                      ;MEMORY PARITY TRAP VECTOR
3469  003164  000000              TCONLO: .WORD   0               ;LO BITS OF CALIBRATION TIME CONSTANT
3470  003166  000000              TCONHI: .WORD   0               ;HI BITS OF CALIB TIME CONST
3471  003170  000000              BLWMIN: .WORD   0               ;COUNT OF TIMES BELOW MIN
3472  003172  000000              ABVMX1: .WORD   0               ;COUNT OF FORWARD TIMES ABOVE MAX
3473  003174  000000              ABVMX2: .WORD   0               ;COUNT OF REVERSE TIMES ABOVE MAX
3474  003176  000000              SUMLO1: .WORD   0               ;LO BITS OF FORWARD TIME SUM
3475  003200  000000              SUMHI1: .WORD   0               ;HI BITS OF FORWARD TIME SUM
3476  003202  000000              SUMLO2: .WORD   0               ;LO BITS OF REVERSE TIME SUM
3477  003204  000000              SUMHI2: .WORD   0               ;HI BITS OF REVERSE TIME SUM
3478  003206  000000              SAVPAR: .WORD   0               ;SAVE WORD FOR PAR CONSTANT
3479  003210  000000              SAVWRD: .WORD   0               ;SCRATCH WORD
3480  003212  000000              MINIL1: .WORD   0               ;LO BITS OF MIN MEAS'D TIME (FORWARD)
3481  003214  000000              MINIH1: .WORD   0               ;HI BITS OF MIN MEAS'D TIME (FORWARD)
3482  003216  000000              MAXIL1: .WORD   0               ;LO BITS OF MAX MEAS'D TIME (FORWARD)
3483  003220  000000              MAXIH1: .WORD   0               ;HI BITS OF MAX MEAS'D TIME (FORWARD)
3484  003222  000000              MINIL2: .WORD   0               ;LO BITS OF MIN MEAS'D TIME (REVERSE)
3485  003224  000000              MINIH2: .WORD   0               ;HI BITS OF MIN MEAS'D TIME (REVERSE)
3486  003226  000000              MAXIL2: .WORD   0               ;LO BITS OF MAX MEAS'D TIME (REVERSE)
3487  003230  000000              MAXIH2: .WORD   0               ;HI BITS OF MAX MEAS'D TIME (REVERSE)
3488  003232  000400              BSSOFT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM SOFTWARE
3489  004232  000400              BSFACT: .BLKW   ^D256           ;RECORD OF BAD SECTORS FROM FACTORY
3490  005232  000000  000000  000000  PRVCMD: .WORD   0,0,0,0,0,0     ;PREVIOUS COMMAND STORAGE
3491  005240  000000  000000  000000
3492  005246  000000  000000  000000  COMSTR: .WORD   0,0,0,0,0,0     ;CURRENT COMMAND STORAGE
3493  005254  000000  000000  000000
3494  005262  000000              PRMPLO: .WORD   0               ;PREV. U.B. MAP REG 0
3495  005264  000000              PRMPHO: .WORD   0
3496  005266  000000              CRMPLO: .WORD   0               ;CURRENT U.B. MAP REG 0
3497  005270  000000              CRMPHO: .WORD   0
3498  005272  000102              BUFFO:  .BLKW   ^D66            ;OUTPUT BUFFER 1
3499  005476  000000              LOWOCT: .WORD   0               ;LOW 16 BITS OF CONVERTED BINARY NUMBER
3500  005500  000000              HIGOCT: .WORD   0               ;HIGH BITS OF CONVERTED BINARY NO.
3501  005502  000000              BUFPRT: .WORD   0               ;BUFFER POINTER
3502  005504  000000              RECODE: .WORD   0               ;RECOVERY CODE WORD
3503  005506  000000              ERRCOM: .WORD   0               ;ERROR COMMAND
```

```
3504  005510  000000          DRIVE:  .WORD   0               ;NO.  OF  DRIVE IN USE
3505  005512  000000          STALLS: .WORD   0               ;CURRENT NO. OF UNIT STALLS TO APPLY
3506  005514  000000          CYLNDR: .WORD   0               ;CURRENT CYLINDER NUMBER
3507  005516  000000          FS:     .WORD   0               ;FIRST SECTOR LIMIT
3508  005520  000000          LS:     .WORD   0               ;LAST SECTOR LIMIT
3509  005522  000000          NCYL1:  .WORD   0               ;NEXT CYL SCRATCH WORD
3510  005524  000000          NCYL2:  .WORD   0               ;NEXT CYL SCRATCH WORD
3511  005526  000000          OFINUS: .WORD   0               ;OFFSET IN USE
3512  005530  000000          TRACK:  .WORD   0               ;TRACK IN USE
3513  005532  000000          INTCHR: .WORD   0               ;TTY INTERRUPT INPUT WORD
3514  005534  000000          SELECT: .WORD   0               ;ANY BIT SET IN LOW BYTE GIVES AUTO-SELECTED DRIVE NO.
3515  005536  000000          CNLINE: .WORD   0               ;ANY BIT SET IN LOW BYTE INDICATES A DRIVE WAS ON-LINE
3516  005540  000000          NEWON:  .WORD   0               ;NEW LOOK AT ONLINE DRIVES
3517  005542  000000          SCRACH: .WORD   0               ;ALL-PURPOSE SCRATCH WORD
3518  005544  000000          PATRN:  .WORD   0               ;DATA PATTERN LIST WORD
3519  005546  000000          XXDPAD: .WORD   0               ;STARTING ADDRESS OF XXDP LOADER
3520  005550  000002          XDPSAV: .BLKW   2               ;XXDP PHYSICAL SAVE ADDRESS
3521  005554  000000          PLOFST: .WORD   0               ;(+) OFFSET VALUE
3522  005556  000000          NGOFST: .WORD   0               ;(-) OFFSET VALUE
3523  005560  000005          SAVPRS: .BLKW   5               ;SAVE INITIAL PARAMS FOR XFER
3524  005572  000000          WDSXFR: .WORD   0               ;NO. OF WORDS ACTUALLY XFERRED
3525  005574  000000          FINCYL: .WORD   0               ;FINAL CYLINDER ADRS
3526  005576     000          FINTRK: .BYTE   0               ;FINAL TRACK ADRS
3527  005577     000          FINSEC: .BYTE   0               ;FINAL SECTOR ADRS
3528  005600  000000          LASTWC: .WORD   0               ;ACTUAL FINAL WC
3529  005602  000002          PMA:    .BLKW   2               ;PARTIAL TRANSFER MEMORY START ADRS
3530
3531
3532
3533                          ;LIST OF DESIRED DRIVES TO TEST (IF BYTE=0, DON'T TEST DRIVE)
3534  005606                  DRVLST:
3535  005606     001                  .BYTE   1               ;DRIVE 0
3536  005607     001                  .BYTE   1               ;DRIVE 1
3537  005610     001                  .BYTE   1               ;DRIVE 2
3538  005611     001                  .BYTE   1               ;DRIVE 3
3539  005612     001                  .BYTE   1               ;DRIVE 4
3540  005613     001                  .BYTE   1               ;DRIVE 5
3541  005614     001                  .BYTE   1               ;DRIVE 6
3542  005615     001                  .BYTE   1               ;DRIVE 7
3543
3544                          ;LIST OF DRIVE TYPES  0=RK06, NON 0=RK07
3545  005616                  DTYLST:
3546  005616     000                  .BYTE   0               ;DRV 0
3547  005617     000                  .BYTE   0               ;DRV 1
3548  005620     000                  .BYTE   0               ;DRV 2
3549  005621     000                  .BYTE   0               ;DRV 3
3550  005622     000                  .BYTE   0               ;DRV 4
3551  005623     000                  .BYTE   0               ;DRV 5
3552  005624     000                  .BYTE   0               ;DRV 6
3553  005625     000                  .BYTE   0               ;DRV 7
3554
3555
3556
3557                          ;LIST OF ITERATION COUNTS FOR DESIRED TESTS (IF=0, TEST NOT RUN)
3558                          ;MAXIMUM ALLOWABLE NUMBER OF ITERATIONS = 77776 (OCT)
3559  005626                  TSTLST:
```

```
3560  005626  000010                  .WORD   10              ;TEST 01
3561  005630  000200                  .WORD   200             ;TEST 02
3562  005632  000010                  .WORD   10              ;TEST 03
3563  005634  000010                  .WORD   10              ;TEST 04
3564  005636  000002                  .WORD   2               ;TEST 05
3565  005640  000002                  .WORD   2               ;TEST 06
3566  005642  000002                  .WORD   2               ;TEST 07
3567  005644  000400                  .WORD   400             ;TEST 10
3568  005646  000500                  .WORD   500             ;TEST 11
3569  005650  000002                  .WORD   2               ;TEST 12
3570  005652  000001                  .WORD   1               ;TEST 13
3571  005654  000001                  .WORD   1               ;TEST 14
3572  005656  000001                  .WORD   1               ;TEST 15
3573  005660  000001                  .WORD   1               ;TEST 16
3574  005662  000010                  .WORD   10              ;TEST 17
3575  005664  000010                  .WORD   10              ;TEST 20
3576  005666  000001                  .WORD   1               ;TEST 21
3577
3578
3579
3580                          ;LIST OF DEFAULT ITERATION COUNTS FOR ALL TESTS
3581  005670                  DFLTST:
3582  005670  000010                  .WORD   10              ;TEST 01
3583  005672  000200                  .WORD   200             ;TEST 02
3584  005674  000010                  .WORD   10              ;TEST 03
3585  005676  000010                  .WORD   10              ;TEST 04
3586  005700  000002                  .WORD   2               ;TEST 05
3587  005702  000002                  .WORD   2               ;TEST 06
3588  005704  000002                  .WORD   2               ;TEST 07
3589  005706  000400                  .WORD   400             ;TEST 10
3590  005710  000500                  .WORD   500             ;TEST 11
3591  005712  000002                  .WORD   2               ;TEST 12
3592  005714  000001                  .WORD   1               ;TEST 13
3593  005716  000001                  .WORD   1               ;TEST 14
3594  005720  000001                  .WORD   1               ;TEST 15
3595  005722  000001                  .WORD   1               ;TEST 16
3596  005724  000010                  .WORD   10              ;TEST 17
3597  005726  000010                  .WORD   10              ;TEST 20
3598  005730  000001                  .WORD   1               ;TEST 21
3599
3600
3601          000021          NMTSTS=<DFLTST-TSTLST>/2        ;TOTAL NO. OF AUTOMATIC TESTS
3602
3603
3604                          ;OPERATING PARAMETER LIST
3605  005732                  PRMLST:
3606  005732  000000          FC:     .WORD   0               ;FIRST CYLINDER
3607  005734  000000          LC:     .WORD   0               ;LAST CYLINDER. 632 FOR RK06, 1456 FOR RK07
3608  005736  000001          IC:     .WORD   1               ;CYLINDER INCREMENT
3609  005740  000000          FT:     .WORD   0               ;FIRST TRACK
3610  005742  000002          LT:     .WORD   2               ;LAST TRACK
3611  005744  000001          IT:     .WORD   1               ;TRACK INCREMENT
3612  005746  000000          S0:     .WORD   0               ;FIRST SECTOR IF 20(DEC) SECTOR FMT
3613  005750  000023          S1:     .WORD   23              ;LAST SECTOR IF 20(DEC) SECTOR FMT
3614  005752  000000          S2:     .WORD   0               ;FIRST SECTOR IF 22(DEC) SECTOR FMT
3615  005754  000025          S3:     .WORD   25              ;LAST SECTOR IF 22(DEC) SECTOR FMT
```

```
3616  005756  000001                IS:     .WORD   1               ;SECTOR INCREMENT
3617  005760  000000                PT:     .WORD   0               ;DATA PATTERN SELECT WORD
3618  005762  063520                MA:     .WORD   RWBUF           ;LO BITS OF PHYS. MEM. ADDR. (BITS 0-15)
3619  005764  000000                        .WORD   0               ;HI BITS OF PHYS. MEM. ADDR. (BITS 16-21)
3620  005766  000403                WC:     .WORD   403             ;WORD COUNT (IF WC=0, WRD CNT IS 65,536 DEC)
3621  005770  000000                CS:     .WORD   0               ;CONTROL SWITCH WORD
3622  005772  000000                ST:     .WORD   0               ;NUMBER OF UNIT STALLS
3623  005774  001000                SM:     .WORD   1000            ;MAX. STALLS, TEST 11
3624
3625
3626                                ;DEFAULT OPERATING PARAMETER VALUES
3627  005776                        PRDFLT:
3628  005776  000000                        .WORD   0               ;FC DEFAULT
3629  006000  000000                        .WORD   0               ;LC DEFAULT
3630  006002  000001                        .WORD   1               ;IC DEFAULT
3631  006004  000000                        .WORD   0               ;FT DEFAULT
3632  006006  000002                        .WORD   2               ;LT DEFAULT
3633  006010  000001                        .WORD   1               ;IT DEFAULT
3634  006012  000000                        .WORD   0               ;S0 DEFAULT
3635  006014  000023                        .WORD   23              ;S1 DEFAULT
3636  006016  000000                        .WORD   0               ;S2 DEFAULT
3637  006020  000025                        .WORD   25              ;S3 DEFAULT
3638  006022  000001                        .WORD   1               ;IS DEFAULT
3639  006024  000000                        .WORD   0               ;PT DEFAULT
3640  006026  063520                        .WORD   RWBUF           ;LOW MA (0-15) DEFAULT
3641  006030  000000                        .WORD   0               ;HIGH MA (16-21) DEFAULT
3642  006032  000403                        .WORD   403             ;WC DEFAULT
3643  006034  000000                        .WORD   0               ;CS DEFAULT
3644  006036  000000                        .WORD   0               ;ST DEFAULT
3645  006040  001000                        .WORD   1000            ;SM DEFAULT
3646
3647
3648
3649                                ;OPERATING PARAMETER VALUE LOW AND HIGH LIMITS
3650  006042                        PRMLIM:
3651  006042  000000                        .WORD   0               ;FC LIMITS
3652  006044  000000                        .WORD   0
3653  006046  000000                        .WORD   0               ;LC LIMITS
3654  006050  000000                        .WORD   0
3655  006052  000001                        .WORD   1               ;IC LIMITS
3656  006054  000000                        .WORD   0
3657  006056  000000                        .WORD   0               ;FT LIMITS
3658  006060  000002                        .WORD   2
3659  006062  000000                        .WORD   0               ;LT LIMITS
3660  006064  000002                        .WORD   2
3661  006066  000001                        .WORD   1               ;IT LIMITS
3662  006070  000002                        .WORD   2
3663  006072  000000                        .WORD   0               ;S0 LIMITS
3664  006074  000023                        .WORD   23
3665  006076  000000                        .WORD   0               ;S1 LIMITS
3666  006100  000023                        .WORD   23
3667  006102  000000                        .WORD   0               ;S2 LIMITS
3668  006104  000025                        .WORD   25
3669  006106  000000                        .WORD   0               ;S3 LIMITS
3670  006110  000025                        .WORD   25
3671  006112  000001                        .WORD   1               ;IS LIMITS
```

```
3672  006114  000025                  .WORD   25
3673  006116  000000                  .WORD   0               ;PT LIMITS
3674  006120  177777                  .WORD   177777
3675  006122  063520                  .WORD   RWBUF           ;MA LOWER LIMIT
3676  006124  000000                  .WORD   0
3677  006126  157776          MAHILM: .WORD   157776          ;MA UPPER LIMIT
3678  006130  000077                  .WORD   77
3679  006132  000000                  .WORD   0               ;WC LIMITS
3680  006134  177777                  .WORD   177777
3681  006136  000000                  .WORD   0               ;CS LIMITS
3682  006140  000070                  .WORD   000070
3683  006142  000000                  .WORD   0               ;ST LIMITS
3684  006144  177777                  .WORD   177777
3685  006146  000000                  .WORD   0               ;SM LIMITS
3686  006150  177777                  .WORD   177777
3687
3688
3689
3690                                  ;ASCII PARAMETER MNEMONICS
3691  006152                          PRMNEM:
3692  006152  041506                  .ASCII  /FC/
3693  006154  041514                  .ASCII  /LC/
3694  006156  041511                  .ASCII  /IC/
3695  006160  052106                  .ASCII  /FT/
3696  006162  052114                  .ASCII  /LT/
3697  006164  052111                  .ASCII  /IT/
3698  006166  030123                  .ASCII  /S0/
3699  006170  030523                  .ASCII  /S1/
3700  006172  031123                  .ASCII  /S2/
3701  006174  031523                  .ASCII  /S3/
3702  006176  051511                  .ASCII  /IS/
3703  006200  052120                  .ASCII  /PT/
3704  006202  040515                  .ASCII  /MA/
3705  006204  000000                  .WORD   0               ;TABLE FILLER FOR MA
3706  006206  041527                  .ASCII  /WC/
3707  006210  051503                  .ASCII  /CS/
3708  006212  052123                  .ASCII  /ST/
3709  006214  046523                  .ASCII  /SM/
3710
3711
3712
3713                                  ;DATA PATTERN 00
3714                                  ;       HI-LO FREQ. MIX
3715  006216                          PAT00:
3716  006216  177777                  .WORD   177777
3717  006220  177777                  .WORD   177777
3718  006222  177777                  .WORD   177777
3719  006224  052525                  .WORD   052525
3720  006226  052525                  .WORD   052525
3721  006230  052525                  .WORD   052525
3722  006232  177777                  .WORD   177777
3723  006234  177777                  .WORD   177777
3724  006236  052525                  .WORD   052525
3725  006240  052525                  .WORD   052525
3726  006242  177777                  .WORD   177777
3727  006244  052525                  .WORD   052525
```

```
 3728  006246  177252                    177252
 3729  006250  177252                    177252
 3730  006252  172765                    172765
 3731  006254  172765                    172765
 3732
 3733
 3734
 3735                                ;DATA PATTERN 01
 3736                                ;       HI FREQ. PHASE MIX
 3737  006256                        PAT01:
 3738  006256  000000                        000000
 3739  006260  000000                        000000
 3740  006262  000000                        000000
 3741  006264  177777                        177777
 3742  006266  177777                        177777
 3743  006270  177777                        177777
 3744  006272  000000                        000000
 3745  006274  000000                        000000
 3746  006276  177777                        177777
 3747  006300  177777                        177777
 3748  006302  000000                        000000
 3749  006304  177777                        177777
 3750  006306  000000                        000000
 3751  006310  177777                        177777
 3752  006312  000000                        000000
 3753  006314  177777                        177777
 3754
 3755
 3756
 3757                                ;DATA PATTERN 02
 3758                                ;       LO FREQ. PHASE MIX
 3759  006316                        PAT02:
 3760  006316  052525                        052525
 3761  006320  052525                        052525
 3762  006322  052525                        052525
 3763  006324  125252                        125252
 3764  006326  125252                        125252
 3765  006330  125252                        125252
 3766  006332  052525                        052525
 3767  006334  052525                        052525
 3768  006336  125252                        125252
 3769  006340  125252                        125252
 3770  006342  052525                        052525
 3771  006344  125252                        125252
 3772  006346  052525                        052525
 3773  006350  125252                        125252
 3774  006352  052525                        052525
 3775  006354  125252                        125252
 3776
 3777
 3778
 3779                                ;DATA PATTERN 03
 3780                                ;       MAX. PRECOMP. PHASE MIX
 3781  006356                        PAT03:
 3782  006356  133333                        133333
 3783  006360  066666                        066666
```

```
3784  006362  155555                  155555
3785  006364  155555                  155555
3786  006366  133333                  133333
3787  006370  066666                  066666
3788  006372  066666                  066666
3789  006374  155555                  155555
3790  006376  155555                  155555
3791  006400  133333                  133333
3792  006402  133333                  133333
3793  006404  133333                  133333
3794  006406  133333                  133333
3795  006410  133333                  133333
3796  006412  133333                  133333
3797  006414  133333                  133333
3798
3799
3800
3801                                  ;DATA PATTERN 04
3802                                  ;       ROTATING BOUNDARY PULSE PRECOMP.
3803  006416                          PAT04:
3804  006416  121105                  121105
3805  006420  150442                  150442
3806  006422  064221                  064221
3807  006424  132110                  132110
3808  006426  055044                  055044
3809  006430  026422                  026422
3810  006432  013211                  013211
3811  006434  105504                  105504
3812  006436  042642                  042642
3813  006440  021321                  021321
3814  006442  110550                  110550
3815  006444  044264                  044264
3816  006446  022132                  022132
3817  006450  011055                  011055
3818  006452  104426                  104426
3819  006454  042213                  042213
3820
3821
3822
3823                                  ;DATA PATTERN 05
3824                                  ;       ROTATING CELL PULSE PRECOMP.
3825  006456                          PAT05:
3826  006456  026455                  026455
3827  006460  113226                  113226
3828  006462  045513                  045513
3829  006464  122645                  122645
3830  006466  151322                  151322
3831  006470  064551                  064551
3832  006472  132264                  132264
3833  006474  055132                  055132
3834  006476  026455                  026455
3835  006500  113226                  113226
3836  006502  045513                  045513
3837  006504  122645                  122645
3838  006506  151322                  151322
3839  006510  064551                  064551
```

```
                                        132264
                                        055132


                        ;DATA PATTERN 06
                        ;       ALL ZEROS
006516                  PAT06:
006516  000000                  000000
006520  000000                  000000
006522  000000                  000000
006524  000000                  000000
006526  000000                  000000
006530  000000                  000000
006532  000000                  000000
006534  000000                  000000
006536  000000                  000000
006540  000000                  000000
006542  000000                  000000
006544  000000                  000000
006546  000000                  000000
006550  000000                  000000
006552  000000                  000000
006554  000000                  000000


                        ;DATA PATTERN 07
                        ;       ALL ONES
006556                  PAT07:
006556  177777                  177777
006560  177777                  177777
006562  177777                  177777
006564  177777                  177777
006566  177777                  177777
006570  177777                  177777
006572  177777                  177777
006574  177777                  177777
006576  177777                  177777
006600  177777                  177777
006602  177777                  177777
006604  177777                  177777
006606  177777                  177777
006610  177777                  177777
006612  177777                  177777
006614  177777                  177777


                        ;DATA PATTERN 08
                        ;       SHIFTED 1 IN FIELD OF ZEROS
006616                  PAT08:
006616  000001                  000001
006620  000002                  000002
006622  000004                  000004
006624  000010                  000010
006626  000020                  000020
006630  000040                  000040
```

```
3896  006632  000100                    000100
3897  006634  000200                    000200
3898  006636  000400                    000400
3899  006640  001000                    001000
3900  006642  002000                    002000
3901  006644  004000                    004000
3902  006646  010000                    010000
3903  006650  020000                    020000
3904  006652  040000                    040000
3905  006654  100000                    100000
3906
3907
3908
3909                              ;DATA PATTERN 09
3910                              ;       SHIFTED 0 IN FIELD OF ONES
3911  006656                      PAT09:
3912  006656  177776                    177776
3913  006660  177775                    177775
3914  006662  177773                    177773
3915  006664  177767                    177767
3916  006666  177757                    177757
3917  006670  177737                    177737
3918  006672  177677                    177677
3919  006674  177577                    177577
3920  006676  177377                    177377
3921  006700  176777                    176777
3922  006702  175777                    175777
3923  006704  173777                    173777
3924  006706  167777                    167777
3925  006710  157777                    157777
3926  006712  137777                    137777
3927  006714  077777                    077777
3928
3929
3930
3931                              ;DATA PATTERN 10
3932                              ;       ALTERNATING 0-1
3933  006716                      PAT10:
3934  006716  052525                    052525
3935  006720  052525                    052525
3936  006722  052525                    052525
3937  006724  052525                    052525
3938  006726  052525                    052525
3939  006730  052525                    052525
3940  006732  052525                    052525
3941  006734  052525                    052525
3942  006736  052525                    052525
3943  006740  052525                    052525
3944  006742  052525                    052525
3945  006744  052525                    052525
3946  006746  052525                    052525
3947  006750  052525                    052525
3948  006752  052525                    052525
3949  006754  052525                    052525
3950
3951
```

```
                                        ;DATA PATTERN 11
                                        ;       ALTERNATING 1-0
3955  006756                            PAT11:
3956  006756  125252                           125252
3957  006760  125252                           125252
3958  006762  125252                           125252
3959  006764  125252                           125252
3960  006766  125252                           125252
3961  006770  125252                           125252
3962  006772  125252                           125252
3963  006774  125252                           125252
3964  006776  125252                           125252
3965  007000  125252                           125252
3966  007002  125252                           125252
3967  007004  125252                           125252
3968  007006  125252                           125252
3969  007010  125252                           125252
3970  007012  125252                           125252
3971  007014  125252                           125252
3972
3973
3974
3975                                    ;DATA PATTERN 12
3976                                    ;       SHIFTING ZEROS AND ONES
3977  007016                            PAT12:
3978  007016  000001                           000001
3979  007020  000003                           000003
3980  007022  000007                           000007
3981  007024  000017                           000017
3982  007026  000037                           000037
3983  007030  000077                           000077
3984  007032  000177                           000177
3985  007034  000377                           000377
3986  007036  000777                           000777
3987  007040  001777                           001777
3988  007042  003777                           003777
3989  007044  007777                           007777
3990  007046  017777                           017777
3991  007050  037777                           037777
3992  007052  077777                           077777
3993  007054  177777                           177777
3994
3995
3996
3997
3998                                    ;DATA PATTERN 13
3999  007056                            ;       COMPOSITE ROTATING
4000  007056  072307                    PAT13:
4001  007060  135143                           072307
4002  007062  156461                           135143
4003  007064  167230                           156461
4004  007066  073514                           167230
4005  007070  035646                           073514
4006  007072  016723                           035646
4007  007074  107351                           016723
                                                107351
```

```
 4008  007076  143564                    143564
 4009  007100  061672                    061672
 4010  007102  030735                    030735
 4011  007104  114356                    114356
 4012  007106  046167                    046167
 4013  007110  123073                    123073
 4014  007112  151453                    151453
 4015  007114  164616                    164616
 4016
 4017
 4018
 4019                              ;DATA PATTERN 14
 4020                              ;       PSEUDO-RANDOM (COMPUTED BY PROGRAM)
 4021  007116                      PAT14:
 4022  007116  000000                      .WORD
 4023  007120  000000                      .WORD
 4024  007122  000000                      .WORD
 4025  007124  000000                      .WORD
 4026  007126  000000                      .WORD
 4027  007130  000000                      .WORD
 4028  007132  000000                      .WORD
 4029  007134  000000                      .WORD
 4030  007136  000000                      .WORD
 4031  007140  000000                      .WORD
 4032  007142  000000                      .WORD
 4033  007144  000000                      .WORD
 4034  007146  000000                      .WORD
 4035  007150  000000                      .WORD
 4036  007152  000000                      .WORD
 4037  007154  000000                      .WORD
 4038
 4039
 4040                              ;USER-DEFINED DATA PATTERN 15
 4041  007156                      PAT15:
 4042  007156  072307                      072307          ;WORD 00
 4043  007160  135143                      135143          ;WORD 01
 4044  007162  156461                      156461          ;WORD 02
 4045  007164  167230                      167230          ;WORD 03
 4046  007166  073514                      073514          ;WORD 04
 4047  007170  035646                      035646          ;WORD 05
 4048  007172  016723                      016723          ;WORD 06
 4049  007174  107351                      107351          ;WORD 07
 4050  007176  143564                      143564          ;WORD 10
 4051  007200  061672                      061672          ;WORD 11
 4052  007202  030735                      030735          ;WORD 12
 4053  007204  114356                      114356          ;WORD 13
 4054  007206  046167                      046167          ;WORD 14
 4055  007210  123073                      123073          ;WORD 15
 4056  007212  151453                      151453          ;WORD 16
 4057  007214  164616                      164616          ;WORD 17
 4058
 4059
 4060
 4061          057467      MINLO1=^D24375                  ;LO BITS OF SPEC'D MIN ROT. LATENCY
 4062          000000      MINHI1=0                        ;HI BITS OF SPEC'D MIN ROT. LATENCY
 4063          062031      MAXLO1=^D25625                  ;LO BITS OF SPEC'D MAX ROT. LATENCY
```

```
4064            000000                  MAXHI1=0                        ;HI BITS OF SPEC'D MAX ROT. LATENCY
4065                                    ;MAXLO2=^D8000                  ;LO BITS OF SPEC'D MAX 1 CYL SEEK TIME
4066                                    ;MAXHI2=0                       ;HI BITS OF SPEC'D MAX 1 CYL SEEK TIME
4067  007216    000000                  MAXLO2: .WORD   0       ;LOAD FROM NEW DRIVE ROUTINE
4068  007220    000000                  MAXHI2: .WORD   0               ;LOAD FROM NEWDRIVE ROUTINE
4069            112160                  MAXLO3=^D38000                  ;LO BITS OF SPEC'D MAX AVG SEEK TIME
4070            000000                  MAXHI3=0                        ;HI BITS OF SPEC'D MAX AVG SEEK TIME
4071                                    ;MAXLO4=^D9464                  ;LO BITS OF SPEC'D MAX 410 CYL SEEK TIME
4072                                    ;MAXHI4=1                        HI BITS OF SPEC'D MAX 410 CYL SEEK TIME
4073  007222    000000                  MAXLO4: .WORD   0       ;LOAD FROM THE DEWDRIVE ROUTINE
4074  007224    000000                  MAXHI4: .WORD   0       ;LOAD FROM THE NEWDRIVE ROUTINE
4075            002734                  RNDSHT=^D1500                   ;SHORT RANDOM SEEKS = 1500(10)
4076            016514                  RNDLNG=^D7500                   ;LONG RANDOM SEEKS = 7500(10)
4077            000002                  LSTTRK=2                        ;LAST TRACK  = 2
4078            000365                  ALNCYL=365                      ;ALIGNMENT CYLINDER =245(10)
4079            000002                  BSERR=BIT1                      ;BSE ERROR
4080            000004                  HVRCER=BIT2                     ;HVRC ERROR
4081            000010                  OPIERR=BIT3                     ;OPI ERROR
4082            000020                  DCKERR=BIT4                     ;DATA CHECK ERROR
4083            000040                  ECCNC=BIT5                      ;ECC NON-CORRECTABLE
4084            000100                  WCERR=BIT6                      ;WRITE CHECK ERROR
4085            000200                  ABORT=BIT7                      ;ABORT
4086            000400                  LEV2ER=BIT8                     ;LEVEL TWO ERROR
4087            001000                  BADSEC=BIT9                     ;BAD SECTOR FLAG
4088            002000                  TWOTOS=BIT10                    ;TWO TIME OUTS
4089            004000                  RCLREQ=BIT11                    ;RECALIBRATE REQUIRED
4090            010000                  DRNAVL=BIT12                    ;DRIVE NOT RELSD BY OTHER PORT
4091            100000                  ANYDER=BIT15                    ;ANY ERROR DETECTED FLAG
4092
4093  007226    005015  055103  033'22  DZR6M:  .ASCIZ  <15><12>/CZR6ME0/
4094  007234    042515  000060
4095  007240    005015  044103  047101  XXDPMG: .ASCII  <CR><LF>/CHANGE XXDP PACK/
4096  007246    042507  054040  042130
4097  007254    020120  040520  045503
4098  007262    005015  046103  040505          .ASCIZ  <CR><LF>/CLEAR LOC 40,RESTART PROGRAM/
4099  007270    020122  047514  020103
4100  007276    030064  051054  051505
4101  007304    040524  052122  050040
4102  007312    047522  051107  046501
4103  007320       000
4104  007321       040  020055  045522  SUBVER: .ASCIZ  & - RK611/06 SS VFY&
4105  007326    030466  027461  033060
4106  007334    051440  020123  043126
4107  007342    000131
4108  007344    006461  000012          PART1:  .ASCIZ  /1/<15><12>
4109  007350    006462  000012          PART2:  .ASCIZ  /2/<15><12>
4110  007354    005015  046012  051501  LSTMEM: .ASCIZ  <15><12><12>/LAST PHYS MEM ADR = /
4111  007362    020124  044120  051531
4112  007370    046440  046505  040440
4113  007376    051104  036440  000040
4114  007404    053117  051105  040514  OVLODR: .ASCIZ  /OVERLAY LOADER ? (Y OR N) * /
4115  007412    020131  047514  042101
4116  007420    051105  037440  024040
4117  007426    020131  051117  047040
4118  007434    020051  020052     000
4119  007441       015  050012  051101  PRMINP: .ASCIZ  <15><12>/PARAM INPUT MODE/<15><12><12>
```

```
4120  007446  046501  044440  050116
4121  007454  052125  046440  042117
4122  007462  006505  005012     000
4123  007467     122  030113  026466  RKBADR: .ASCIZ  /RK06-07 BUS ADR = /
4124  007474  033460  041040  051525
4125  007502  040440  051104  036440
4126  007510  000040
4127  007512  045522  033060  030055  RKVADR: .ASCIZ  /RK06-07 VEC ADR = /
4128  007520  020067  042526  020103
4129  007526  042101  020122  020075
4130  007534     000
4131  007535     122  030113  026466  RKPRTY: .ASCIZ  /RK06-07 PRIOR =/
4132  007542  033460  050040  044522
4133  007550  051117  036440     000
4134  007555     123  051127  036440  SWRMSG: .ASCIZ  /SWR = /
4135  007562  000040
4136  007564  020040  042516  020127  NEWMSG: .ASCIZ  /  NEW = /
4137  007572  020075     000
4138  007575     015  042012  053122  DRVSEQ: .ASCIZ  <15><12>/DRV(S) = /
4139  007602  051450  020051  020075
4140  007610     000
4141  007611     104  044522  042526  BADDRV: .ASCIZ  /DRIVE   /
4142  007616  020040  000040
4143  007622  047516  026516  054105  NXDRIV: .ASCIZ  /NON-EXIST/<15><12>
4144  007630  051511  006524  000012
4145  007636  047516  020124  042122  NTREDY: .ASCIZ  /NOT RDY/<15><12>
4146  007644  006531  000012
4147  007650  051127  026524  047514  WRTLOK: .ASCIZ  /WRT-LOCK/<15><12>
4148  007656  045503  005015     000
4149  007663     114  040517  042504  ALNPAK: .ASCIZ  /LOADED WITH ALIGN PACK/<15><12>
4150  007670  020104  044527  044124
4151  007676  040440  044514  047107
4152  007704  050040  041501  006513
4153  007712  000012
4154  007714  043111  054040  042130  REPLPK: .ASCIZ  /IF XXDP PACK ON DRV 0, TYPE "Y <CR>", &  REPLACE IT : /
4155  007722  020120  040520  045503
4156  007730  047440  020116  051104
4157  007736  020126  026060  052040
4158  007744  050131  020105  054442
4159  007752  036040  051103  021076
4160  007760  020054  020046  051040
4161  007766  050105  040514  042503
4162  007774  044440  020124  020072
4163  010002     000
4164  010003     015  005012  025052  NODRTS: .ASCII  <15><12><12>/** NO DRVS TO TEST/<15><12>
4165  010010  047040  020117  051104
4166  010016  051526  052040  020117
4167  010024  042524  052123  005015
4168  010032  051120  051505  020123  CNTRDY: .ASCIZ  /PRESS "CONT" WHEN RDY/<15><12>
4169  010040  041442  047117  021124
4170  010046  053440  042510  020116
4171  010054  042122  006531  000012
4172  010062  040510  052114  051040  HLTRQD: .ASCIZ  /HALT REQ/<15><12>
4173  010070  050505  005015     000
4174  010075     104  053122  030040  DROXDP: .ASCIZ  /DRV 0 IS LOAD MEDIUM/<15><12>
4175  010102  044440  020123  047514
```

```
4176  010110  042101  046440  042105
4177  010116  052511  006515  000012
4178  010124  005015  052012  020117  ALDRVS: .ASCIZ  <15><12><12>/TO TEST ALL DRVS TYPE "A" <CR>, ELSE <CR>/<15><12>/* /
4179  010132  042524  052123  040440
4180  010140  046114  042040  053122
4181  010146  020123  054524  042520
4182  010154  021040  021101  036040
4183  010162  051103  026076  042440
4184  010170  051514  020105  041474
4185  010176  037122  005015  020052
4186  010204     000
4187  010205     015  046012  036440  TSTMDS: .ASCII  <15><12>/L = LIST TESTS/<15><12>
4188  010212  046040  051511  020124
4189  010220  042524  052123  006523
4190  010226     012
4191  010227     103  036440  041440          .ASCII  /C = CHANGE TESTS/<15><12>
4192  010234  040510  043516  020105
4193  010242  042524  052123  006523
4194  010250     012
4195  010251     111  036440  044440          .ASCIZ  /I = INPUT PARAMS, RUN TESTS/<15><12>
4196  010256  050116  052125  050040
4197  010264  051101  046501  026123
4198  010272  051040  047125  052040
4199  010300  051505  051524  005015
4200  010306     000
4201  010307     015  042412  052116  ENTLCI: .ASCIZ  <15><12>/ENTER L,C, OR I/<15><12>/* /
4202  010314  051105  046040  041454
4203  010322  020054  051117  044440
4204  010330  005015  020052     000
4205  010335     124  020117  042504  DFTEST: .ASCIZ  /TO DEFAULT TESTS TYPE D <CR>, ELSE <CR>/<15><12>/* /
4206  010342  040506  046125  020124
4207  010350  042524  052123  020123
4208  010356  054524  042520  042040
4209  010364  036040  051103  026076
4210  010372  042440  051514  020105
4211  010400  041474  037122  005015
4212  010406  020052     000
4213  010411     015  052012  051505  TLSTHD: .ASCIZ  <15><12>/TEST     ITERATIONS/<15><12>
4214  010416  020124  020040  020040
4215  010424  052111  051105  052101
4216  010432  047511  051516  005015
4217  010440     000
4218  010441     015  052012  036440  EXPLAN: .ASCII  <15><12>/T = TYPE PARAM LIST/<15><12>
4219  010446  052040  050131  020105
4220  010454  040520  040522  020115
4221  010462  044514  052123  005015
4222  010470  020117  020075  050117          .ASCII  /O = OPEN PARAM LIST/<15><12>
4223  010476  047105  050040  051101
4224  010504  046501  046040  051511
4225  010512  006524     012
4226  010515     123  036440  051440          .ASCII  /S = SET INDIVIDUAL PARAM/<15><12>
4227  010522  052105  044440  042116
4228  010530  053111  042111  040525
4229  010536  020114  040520  040522
4230  010544  006515     012
4231  010547     122  036440  051040          .ASCIZ  /R = RUN TESTS/<15><12>
```

```
4232  010554  047125  052040  051505
4233  010562  051524  005015     000
4234  010567     015  042412  052116  PARMDE: .ASCIZ  <15><12>/ENTER T,O,S, OR R/<15><12>
4235  010574  051105  052040  047454
4236  010602  051454  020054  051117
4237  010610  051040  005015     000
4238  010615     015  025012  042040  DUACES: .ASCIZ  <15><12>/* DUAL-PORT DATA TEST */<15><12>
4239  010622  040525  026514  047520
4240  010630  052122  042040  052101
4241  010636  020101  042524  052123
4242  010644  025040  005015     000
4243  010651     115  054101  053440  MAWRDC: .ASCIZ  /MAX WORD CT = /
4244  010656  051117  020104  052103
4245  010664  036440  000040
4246  010670  025052  020040  030123  SECNL1: .ASCII  /**  S0>S1/
4247  010676  051476     061
4248  010701     040  047516  020124  NOTALD: .ASCIZ  / NOT ALLOWED/<15><12>
4249  010706  046101  047514  042527
4250  010714  006504  000012
4251  010720  025052  020040  031123  SECNL2: .ASCIZ  /**  S2>S3/
4252  010726  051476  000063
4253  010732  025052  020040  052106  TRKNLM: .ASCIZ  /**  FT>LT/
4254  010740  046076  000124
4255  010744  025052  020040  041527  WC2BIG: .ASCIZ  /**  WC OR MA TOO LARGE/<15><12>
4256  010752  047440  020122  040515
4257  010760  052040  047517  046040
4258  010766  051101  042507  005015
4259  010774     000
4260  010775     124  020117  042504  DFQUES: .ASCIZ  /TO DEFAULT ALL PARAMS, TYPE D <CR>, ELSE <CR>/<15><12>/* /
4261  011002  040506  046125  020124
4262  011010  046101  020114  040520
4263  011016  040522  051515  020054
4264  011024  054524  042520  042040
4265  011032  036040  051103  026076
4266  011040  042440  051514  020105
4267  011046  041474  037122  005015
4268  011054  020052     000
4269  011057     015  052412  042523  PFIFTN: .ASCIZ  <15><12>/USER-DEF PATT 15 :/<15><12>
4270  011064  026522  042504  020106
4271  011072  040520  052124  030440
4272  011100  020065  006472  000012
4273  011106  005015  047515  020104  SELP15: .ASCIZ  <15><12>/MOD PATT 15 :/<15><12>
4274  011114  040520  052124  030440
4275  011122  020065  006472  000012
4276  011130  047524  046440  042117  MDFY15: .ASCIZ  /TO MOD PATT 15, TYPE M <CR>, ELSE <CR>/<15><12>/* /
4277  011136  050040  052101  020124
4278  011144  032461  020054  054524
4279  011152  042520  046440  036040
4280  011160  051103  026076  042440
4281  011166  051514  020105  041474
4282  011174  037122  005015  020052
4283  011202     000
4284  011203     015  042412  052116  ENTPAS: .ASCIZ  <15><12>/ENIER NO. OF PASSES (1-77777) :/<15><12>/* /
4285  011210  051105  047040  027117
4286  011216  047440  020106  040520
4287  011224  051523  051505  024040
```

```
 4288  011232  026461  033467  033467
 4289  011240  024467  035040  005015
 4290  011246  020052     000
 4291  011251     015  025012  020052  DROPDR: .ASCIZ  <15><12>/**  DROPPING DRV - 20 ERRORS/<15><12>
 4292  011256  042040  047522  050120
 4293  011264  047111  020107  051104
 4294  011272  020126  020055  030062
 4295  011300  042440  051122  051117
 4296  011306  006523  000012
 4297  011312  005015  052012  051505  TSTDRN: .ASCII  <15><12><12>/TESTING DRV /
 4298  011320  044524  043516  042040
 4299  011326  053122     040
 4300  011331     040  005015     000  DRVNO:  .ASCIZ  / /<15><12>
 4301  011335     104  053122  000040  DRIV:   .ASCIZ  /DRV /
 4302  011342  040503  052122  020056  CART:   .ASCIZ  /CART. /
 4303  011350     000
 4304  011351     123  051105  020056  SERNM:  .ASCIZ  /SER. NO.  /
 4305  011356  047516  020056  000040
 4306  011364  005015  040506  052103  FACTBS: .ASCIZ  <15><12>/FACTORY /
 4307  011372  051117  020131     000
 4308  011377     012  047523  052106  SOFTBS: .ASCII  <12>/SOFTWARE /
 4309  011404  040527  042522     040
 4310  011411     102  042101  051440  BDSECT: .ASCIZ  /BAD SECTORS :/<15><12>
 4311  011416  041505  047524  051522
 4312  011424  035040  005015     000
 4313  011431     040  047040  047117  NOFALS: .ASCIZ  /  NONE/
 4314  011436  000105
 4315
 4316                                  ;MESSAGES USED IN TIMING TESTS
 4317  011440  025052  020040  047516  NOCLKS: .ASCII  /**  NO CLOCK/
 4318  011446  041440  047514  045503
 4319  011454  026440  020040  044524  TIMSKP: .ASCIZ  / -  TIMING TESTS WILL BE SKIPPED/<15><12>
 4320  011462  044515  043516  052040
 4321  011470  051505  051524  053440
 4322  011476  046111  020114  042502
 4323  011504  051440  044513  050120
 4324  011512  042105  005015     000
 4325  011517     052  020052  045440  CLKFAL: .ASCIZ  /**  KW11 FAILURE/
 4326  011524  030527  020061  040506
 4327  011532  046111  051125  000105
 4328  011540  005015  047522  020124  ROTIMS: .ASCIZ  <15><12>/ROT TIMES :/<15><12>
 4329  011546  044524  042515  020123
 4330  011554  006472  000012
 4331  011560  005015  053101  020107  AVGSEK: .ASCIZ  <15><12>/AVG SEEK TIMES :/<15><12>
 4332  011566  042523  045505  052040
 4333  0..574  046511  051505  035040
 4334  011602  005015     000
 4335  011605     015  047412  042516  ONECYL: .ASCIZ  <15><12>/ONE CYL SEEK TIMES :/<15><12>
 4336  011612  041440  046131  051440
 4337  011620  042505  020113  044524
 4338  011626  042515  020123  006472
 4339  011634  000012
 4340  011636  005015  040515  020130  MAXSEK: .ASCIZ  <15><12>/MAX SEEK TIMES :/<15><12>
 4341  011644  042523  045505  052040
 4342  011652  046511  051505  035040
 4343  011660  005015     000
```

```
 4344  011663     052  043052  042127  FORWRD: .ASCIZ  /**FWD DIR**/<15><12>
 4345  011670  042040  051111  025052
 4346  011676  005015     000
 4347  011701     052  051052  053105  REVRSE: .ASCIZ  /**REV DIR**/<15><12>
 4348  011706  042040  051111  025052
 4349  011714  005015     000
 4350  011717     115  047111  036440  MINEQ:  .ASCIZ  /MIN = /
 4351  011724  000040
 4352  011726  040515  020130  020075  MAXEQ:  .ASCIZ  /MAX = /
 4353  011734     000
 4354  011735     101  043526  036440  AVGEQ:  .ASCIZ  /AVG = /
 4355  011742  000040
 4356  011744  052440  000123          MICROS: .ASCIZ  / US/
 4357  011750  047440  020106  031061  BELOW:  .ASCIZ  / OF 128 BELOW SPEC'D MIN OF /
 4358  011756  020070  042502  047514
 4359  011764  020127  050123  041505
 4360  011772  042047  046440  047111
 4361  012000  047440  020106     000
 4362  012005     040  043117  030440  ABOVE:  .ASCIZ  / OF 128 ABOVE SPEC'D MAX OF /
 4363  012012  034062  040440  047502
 4364  012020  042526  051440  042520
 4365  012026  023503  020104  040515
 4366  012034  020130  043117  000040
 4367  012042  047440  020106  030464  ABOVE1: .ASCII  / OF 410 (814 FOR RK07) ABOVE SPEC'D MAX OF 8000 US/
 4368  012050  020060  034050  032061
 4369  012056  043040  051117  051040
 4370  012064  030113  024467  040440
 4371  012072  047502  042526  051440
 4372  012100  042520  023503  020104
 4373  012106  040515  020130  043117
 4374  012114  034040  030060  020060
 4375  012122  051525
 4376  012124  033050  030065  026460          .ASCIZ  /(6500-RK07)/<CR><LF>
 4377  012132  045522  033460  006451
 4378  012140  000012
 4379  012142  047440  020106  031061  ABOVE3: .ASCII  / OF 128 ABOVE SPEC'D MAX OF 75000 US/
 4380  012150  020070  041101  053117
 4381  012156  020105  050123  041505
 4382  012164  042047  046440  054101
 4383  012172  047440  020106  032467
 4384  012200  030060  020060  051525
 4385  012206  033450  030063  030060          .ASCIZ  /(73000-RK07)/<CR><LF>
 4386  012214  051055  030113  024467
 4387  012222  005015     000
 4388  012225     040  020040  050123  SPCDMX: .ASCIZ  /   SPEC'D MAX IS 38000 US/<15><12>
 4389  012232  041505  042047  046440
 4390  012240  054101  044440  020123
 4391  012246  034063  030060  020060
 4392  012254  051525  005015     000
 4393  012261     062  031464  032467  LIM1:   .ASCIZ  /24375 US/<15><12>
 4394  012266  052440  006523  000012
 4395  012274  032462  031066  020065  LIM2:   .ASCIZ  /25625 US/<15><12>
 4396  012302  051525  005015     000
 4397
 4398  012307     052  020052  041440  BAD632: .ASCIZ  /**  CANNOT READ BAD SECTOR TRALK!/<15><12>
 4399  012314  047101  047516  020124
```

```
 4400  012322  042522  042101  041040
 4401  012330  042101  051440  041505
 4402  012336  047524  020122  051124
 4403  012344  041501  020513  005015
 4404  012352     000
 4405  012353     136  006503  000012  CNTRLC: .ASCIZ  /^C/<15><12>
 4406  012360  055136  005015     000  CNTRLZ: .ASCIZ  /^Z/<15><12>
 4407  012365     136  006522  000012  CNTRLR: .ASCIZ  /^R/<15><12>
 4408  012372  052536  005015     000  CNTRLU: .ASCIZ  /^U/<15><12>
 4409  012377     136  006507  000012  CNTRLG: .ASCIZ  /^G/<15><12>
 4410  012404  005015  000012          CR2LF:  .ASCIZ  <15><12><12>
 4411  012410  000134                  BKSLSH: .ASCIZ  /\/
 4412  012412  000054                  COMMA:  .ASCIZ  /,/
 4413  012414  020040                  SPACE6: .ASCII  /  /
 4414  012416     040                  SPACE4: .ASCII  / /
 4415  012417     040                  SPACE3: .ASCII  / /
 4416  012420     040                  SPACE2: .ASCII  / /
 4417  012421     040     000          SPACE1: .ASCIZ  / /
 4418          012424                  .EVEN
 4419  012424  020040  000075          PRMBUF: .ASCIZ  /  =/
 4420  012430  047527  042122  000040  WORDSP: .ASCIZ  /WORD /
 4421  012436  036440  000040          EQUALS: .ASCIZ  / = /
 4422  012442  020052     000          PROMPT: .ASCIZ  /* /
 4423  012445     076  000040          PRMPSP: .ASCIZ  /> /
 4424
 4425                                  .EVEN
 4426
 4427
 4428
 4429
 4430  012450  105037  003116  DFSTRT: CLRB    MDFLAG          ;SET FLAG FOR DEFAULT MODE
 4431  012454  105037  003136          CLRB    DULACS          ;CLEAR DUAL-ACCESS FLAG
 4432  012460  105037  003126          CLRB    ERRCNT          ;CLEAR ERROR COUNT FOR RESTARTS
 4433  012464  022737  013660  000042  CMP     #DRVTST,@#42    ;SEE IF EOP RETURN ADRS = DRVTST
 4434  012472  001003                  BNE     4$              ;BR IF NOT DRVTST
 4435  012474  012737  016742  000042  MOV     #NEWPAS,@#42    ;SET RETURN ADRS = NEWPAS
 4436  012502  000414          4$:     BR      CMSTRT          ;PROCEED
 4437
 4438  012504  112737  000001  003136  DASTRT: MOVB    #1,DULACS       ;SET FLAG FOR DUAL-ACCESS DATA TEST
 4439  012512  112737  000001  003116          MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
 4440  012520  000405                  BR      CMSTRT          ;PROCEED
 4441
 4442  012522  112737  000001  003116  PSTART: MOVB    #1,MDFLAG       ;SET FLAG FOR PARAMETER MODE
 4443  012530  105037  003136          CLRB    DULACS          ;CLEAR DUAL-ACCESS TEST FLAG
 4444
 4445  012534  012737  000340  177776  CMSTRT: MOV     #PR7,@#PS       ;BLOCK ALL INTERRUPTS
 4446                                  .SBTTL  INITIALIZE THE COMMON TAGS
 4447                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 4448  012542  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 4449  012546  005026          1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 4450  012550  022706  001140          CMP     #SWR,R6 ;;DONE?
 4451  012554  001374                  BNE     .-6             ;;LOOP BACK IF NO
 4452  012556  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 4453                                  ;;INITIALIZE A FEW VECTORS
 4454  012562  012737  055120  000020  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 4455  012570  012737  000340  000022  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
```

```
 4456  012576  012737  054414  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 4457  012604  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 4458  012612  012737  056226  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 4459  012620  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 4460  012626  012737  054764  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 4461  012634  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 4462  012642  013737  025430  025422          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 4463  012650  005037  001304                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 4464  012654  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 4465  012660  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 4466  012666  012737  012666  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 4467  012674  012737  012674  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 4468                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 4469                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 4470  012702  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 4471  012706  012737  012742  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 4472  012714  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 4473  012722  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 4474  012730  022777  177777  166202          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 4475  012736  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 4476                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
 4477  012740  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
 4478  012742  012716  012750          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 4479  012746  000002                          RTI
 4480  012750  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 4481  012756  012737  000174  001142          MOV     #DISPREG,DISPLAY
 4482  012764  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 4483
 4484  012770  005037  001326                  CLR     $PASS           ;;CLEAR PASS COUNT
 4485  012774  132737  000200  001341          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
 4486  013002  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
 4487  013004  012737  001342  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
 4488  013012                          67$:
 4489  013012  000005                          RESET                   ;CLEAR THE UNIBUS
 4490  013014  012737  000006  000004          MOV     #6,@#ERRVEC     ;SET TIME-OUT VECTOR
 4491  013022  005037  000006                  CLR     @#ERRVEC+2
 4492  013026  112737  000144  001115          MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
 4493  013034  105037  003130                  CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
 4494  013040  112737  000001  003146          MOVB    #1,HLPOVL       ;SET HLP FILE OVLD INDICTR
 4495  013046  104401  007226                  TYPE    ,DZR6M          ;TYPE PROGRAM I.D. FOR PART 1
 4496  013052  104401  007321                  TYPE    ,SUBVER
 4497  013056  104401  007344                  TYPE    ,PART1
 4498  013062  105737  003147                  TSTB    NOTYPE          ;SEE IF OPERATOR NOTE SHOULD BE TYPED
 4499  013066  001004                          BNE     39$             ;BR IF NOT
 4500  013070  104401  063520                  TYPE    ,NOTMSG         ;TYPE OPERATOR NOTE
 4501  013074  105237  003147                  INCB    NOTYPE          ;SET FLAG FOR NEXT TIME
 4502  013100  105737  003136          39$:    TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
 4503  013104  001402                          BEQ     40$             ;BR IF NOT
 4504  013106  104401  010615                  TYPE    ,DUACES         ;TYPE "* DUAL-ACCESS DATA TEST *"
 4505  013112  012737  025626  000060  40$:    MOV     #KBDHDL,@#TKVEC ;LOAD VECTOR FOR TTY KBD
 4506  013120  012737  000200  000062          MOV     #PR4,@#TKVEC+2  ;SET KBD PRIORITY = 4
 4507  013126  013701  003040                  MOV     RKVEC,R1        ;ADDR. OF RK06 VECTOR STORAGE
 4508  013132  012721  045234                  MOV     #1.INTR,(R1)+   ;SET IT TO RK06 HANDLER
 4509  013136  013711  003042                  MOV     RKPRI,(R1)      ;SET RK06 PRIORITY
 4510  013142  012737  026436  000250          MOV     #KTERHD,@#MMVEC ;VECT FOR KT11 FAILURE
 4511  013150  012737  000340  000252          MOV     #PR7,@#MMVEC+2  ;SET PRIOR. = 7 FOR HNDLER
```

```
 4512  013156  105037  003120                    CLRB    TSTING          ;CLEAR "RUNNING TESTS" FLAG
 4513  013162  012737  000000  177776            MOV     #?RO,@#PS       ;ALLOW ALL INTERRUPTS AGAIN
 4514  013170  012701  005232                    MOV     #PRVCMD,R1      ;ZERO OUT PREVIOUS COMMAND
 4515  013174  012700  000006                    MOV     #6,R0           ;SET COUNTER
 4516  013200  005021                  42$:      CLR     (R1)+           ;ZERO A WORD
 4517  013202  005300                            DEC     R0
 4518  013204  001375                            BNE     42$             ;BR IF NOT DONE YET
 4519  013206  005037  005512                    CLR     STALLS          ;DON'T ALLOW STALLS YET
 4520  013212  023727  000042  016742            CMP     @#42,#NEWPAS    ;SEE IF CHAIN MODE
 4521  013220  101002                            BHI     44$             ;BR IF YES
 4522  013222  004737  026316                    JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
 4523  013226  012737  176543  052662  44$:      MOV     #176543,$HINUM  ;INIT. PSEUDO-RANDOM NOS.
 4524  013234  012737  123456  052664            MOV     #123456,$LONUM
 4525                          ;CHECK FOR PRESENCE OF KW11-L OR P CLOCK, AND SET FLAGS
 4526  013242  105037  003132                    CLRB    PCLKF           ;INIT. P CLOCK FLAG
 4527  013246  105037  003133                    CLRB    DOTIM           ;INIT. TIMING TESTS FLAG
 4528  013252  012737  013312  000004            MOV     #6$,@#4         ;SET TIME-OUT ERROR VECTOR
 4529  013260  005777  167666                    TST     @PKS            ;SEE IF P-CLOCK IS PRESENT
 4530  013264  105237  003132                    INCB    PCLKF           ;PRESENT, SET FLAG
 4531  013270  013700  003162                    MOV     PCVEC,R0        ;LOAD KW11-P VECTOR ADDRESS
 4532  013274  012720  032620          4$:       MOV     #CLOCK,(R0)+    ;ADDR OF CLOCK SERVICE ROUTINE
 4533  013300  012710  000340                    MOV     #PR7,(R0)       ;SET CLOCK HANDLER PRIORITY = 7
 4534  013304  105237  003133                    INCB    DOTIM           ;SET FLAG TO ALLOW TIMING TESTS
 4535  013310  000414                            BR      8$              ;BR TO CONTINUE
 4536  013312  022626                  6$:       CMP     (SP)+,(SP)+     ;P-CLK NOT THERE, RESET THE STACK
 4537  013314  012737  013334  000004            MOV     #7$,@#4         ;SET TIME-OUT ERROR VECTOR
 4538  013322  005777  167622                    TST     @LKS            ;SEE IF L-CLK PRESENT
 4539  013326  013700  003160                    MOV     LCVEC,R0        ;LOAD KW11-L VECTOR ADDRESS
 4540  013332  000760                            BR      4$              ;BR TO SET UP L-CLK VECTOR
 4541  013334  022626                  7$:       CMP     (SP)+,(SP)+     ;L-CLK NOT THERE, RESET THE STACK
 4542  013336  104401  011440                    TYPE    ,NOCLKS         ;SAY TIMING TESTS WON'T BE RUN
 4543  013342  012737  000006  000004  8$:       MOV     #6,@#4          ;SET TIME-OUT VECTOR TO LOCATION 6
 4544  013350  004737  026020                    JSR     PC,SIZMEM       ;SIZE MEMORY, FIX MA LIMIT IN PRMLIM
 4545  013354  105737  003135                    TSTB    XDPSVD          ;SEE IF XXDP PREVIOUSLY SAVED
 4546  013360  001404                            BEQ     9$              ;BR IF NOT
 4547  013362  004737  027210                    JSR     PC,GETXDP       ;RESTORE SAVED XXDP
 4548  013366  105037  003135                    CLRB    XDPSVD          ;CLEAR THE FLAG
 4549  013372                          9$:
 4550  013372  122737  000013  000041            CMPB    #13,@#41        ;LOAD FROM XXDP ?
 4551  013400  001003                            BNE     99$             ;BRANCH IF NOT
 4552  013402  104401  007240                    TYPE    ,XXDPMG
 4553  013406  000000                            HALT
 4554  013410                          99$:
 4555  013410  105737  003116                    TSTB    MDFLAG          ;SEE IF DEFAULT MODE
 4556  013414  001031                            BNE     10$             ;BR IF NOT DEFAULT MODE
 4557  013416  013700  001370                    MOV     $VECT1,R0       ;GET APT RK06 VECTOR AND PRTY
 4558  013422  013737  001374  003036            MOV     $BASE,RKBAS     ;GET APT RK06 BASE ADDRESS
 4559  013430  132737  000200  001341            BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
 4560  013436  001005                            BNE     18$             ;BR IF NO SIZING
 4561  013440  012700  120210                    MOV     #AVECT1,R0      ;GET DEFAULT VECTOR AND PRIORITY
 4562  013444  012737  177440  003036            MOV     #ABASE,RKBAS    ;GET DEFAULT BASE ADDRESS
 4563  013452  110037  003040          18$:      MOVB    R0,RKVEC        ;STORE VECTOR
 4564  013456  105037  003041                    CLRB    RKVEC+1         ;CLEAR HI BYTE
 4565  013462  000300                            SWAB    R0              ;GET PRTY INTO BITS 5-7
 4566  013464  042700  177437                    BIC     #177437,R0      ;CLEAR OTHER BITS
 4567  013470  010037  003042                    MOV     R0,RKPRI        ;STORE RK06 PRIORITY
```

```
 4568  013474  000137  014372                    JMP     ALLDRV          ;GO CHECK ALL DRIVES
 4569
 4570
 4571
 4572                                    ;BEGIN PARAMETER INPUT MODE
 4573  013500  104401  007441            10$:    TYPE    ,PRMINP         ;TYPE "PARAMETER INPUT MODE"
 4574
 4575                                    ;OPEN RK06 UNIBUS ADDRESS FOR MODIFICATION
 4576  013504  013746  003036                    MOV     RKBAS,-(SP)     ;PUT OLD VALUE ON STACK
 4577  013510  104401  007467                    TYPE    ,RKBADR         ;TYPE "RK06 BUS ADR = "
 4578  013514  004737  026206                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKBAS VALUE
 4579  013520  012637  003036                    MOV     (SP)+,RKBAS     ;STORE NEW VALUE
 4580                                    ;OPEN RK06 VECTOR ADDRESS FOR MODIFICATION
 4581  013524  013746  003040                    MOV     RKVEC,-(SP)     ;PUT OLD VALUE ON STACK
 4582  013530  104401  007512                    TYPE    ,RKVADR         ;TYPE "RK06 VEC ADR = "
 4583  013534  004737  026206                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKVEC VALUE
 4584  013540  012637  003040                    MOV     (SP)+,RKVEC     ;STORE NEW VALUE
 4585                                    ;OPEN RK06 INTERRUPT HANDLER PRIORITY LEVEL FOR MODIFICATION
 4586  013544  013746  003042            11$:    MOV     RKPRI,-(SP)     ;GET OLD VALUE OF PRIORITY
 4587  013550  006316                            ASL     (SP)            ;GET IT INTO BITS 0-2
 4588  013552  006316                            ASL     (SP)
 4589  013554  006316                            ASL     (SP)
 4590  013556  000316                            SWAB    (SP)
 4591  013560  104401  007535                    TYPE    ,RKPRTY         ;TYPE   "RK06 PRIORITY = "
 4592  013564  004737  026206                    JSR     PC,GETPRM       ;TYPE OLD, GET NEW RKPRI VALUE
 4593  013570  012600                            MOV     (SP)+,R0
 4594  013572  020027  000004                    CMP     R0,#4           ;SEE IF AT LEAST LEVEL 4
 4595  013576  002414                            BLT     12$             ;BR IF NEW VALUE TOO SMALL
 4596  013600  020027  000007                    CMP     R0,#7           ;SEE IF LEVEL 7 OR LESS
 4597  013604  003011                            BGT     12$             ;BR IF NEW VALUE TOO LARGE
 4598  013606  000300                            SWAB    R0              ;GET PRIORITY INTO BITS 5-7
 4599  013610  006200                            ASR     R0
 4600  013612  006200                            ASR     R0
 4601  013614  006200                            ASR     R0
 4602  013616  010037  003042                    MOV     R0,RKPRI        ;STORE NEW VALUE
 4603  013622  104401  001315                    TYPE    ,$CRLF
 4604  013626  000405                            BR      16$
 4605  013630  104401  005272            12$:    TYPE    ,BUFF0          ;ECHO BAD INPUT
 4606  013634  104401  001314                    TYPE    ,$QUES
 4607  013640  000741                            BR      11$             ;GO ASK AGAIN
 4608
 4609  013642  105037  003126            16$:    CLRB    ERRCNT          ;CLEAR ERROR CNT FOR RESTARTS
 4610  013646  012737  013660  000042            MOV     #DRVTST,@#42    ;SET UP DUMP MODE RETURN FROM $EOP
 4611                                                                    ;    UPON COMPLETION OF REQUESTED PASSES
 4612  013654  000137  014450                    JMP     CHKLST          ;GO CHECK STATUS OF MARKED DRIVES
 4613
 4614
 4615
 4616                                    ;;*****************************************************************
 4617                                    .SBTTL  TO - DESIRED DRIVE INPUT ROUTINE
 4618                                    ;*THIS ROUTINE TYPES THE CURRENT LIST OF DRIVES TO TEST,
 4619                                    ;*WITH DRIVE NUMBERS SEPARATED BY COMMAS. THEN, A NEW
 4620                                    ;*CHOICE OF DRIVES IS REQUESTED BY TTY INPUT, AND THESE
 4621                                    ;*DRIVES ARE CHECKED FOR VALIDITY AND PROPER STATUS,
 4622                                    ;*AND IF THEY ARE VALID, THEY ARE LOADED INTO THE DRIVE
 4623                                    ;*LIST (DRVLST).
```

```
4624                                    ;;*****************************************************************
4625
4626  013660                            DRVTST:
4627  013660  012706  001100                   MOV     #STACK,SP       ;RESET THE STACK
4628  013664  005037  005512                   CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
4629  013670  104401  010124                   TYPE    ,ALDRVS         ;ASK IF ALL DRIVES DESIRED
4630  013674  004737  030560                   JSR     PC,RDCHRS       ;READ RESPONSE
4631  013700  013660                           DRVTST                  ;(^C) RETURN ADDRESS
4632  013702  013660                           DRVTST                  ;(^Z) RETURN ADDRESS
4633  013704  013660                           DRVTST                  ;(^U) OR ERROR RETURN ADDRESS
4634  013706  005700                           TST     R0              ;SEE IF NULL INPUT
4635  013710  001413                           BEQ     TELDRV          ;BR IF NULL INPUT
4636  013712  022737  000101  005272           CMP     #'A,BUFFO       ;SEE IF ALL DRIVES REQUESTED
4637  013720  001002                           BNE     4$              ;BR IF NOT ALL DRIVES
4638  013722  000137  014372                   JMP     ALLDRV          ;CHECK ALL DRIVES
4639  013726  104401  005272           4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
4640  013732  104401  001314                   TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
4641  013736  000750                           BR      DRVTST          ;GO ASK AGAIN
4642                                    ;TYPE CURRENT DRIVE LIST
4643  013740  104401  007575           TELDRV: TYPE    ,DRVSEQ         ;TYPE 'DRIVE(S) = ''
4644  013744  005001                           CLR     R1              ;INITIALIZE DRIVE NUMBER
4645  013746  005000                           CLR     R0              ;INIT. COUNT OF LISTED DRIVES
4646  013750  012703  000001                   MOV     #BIT0,R3        ;INIT BIT POINTER
4647  013754  105761  005606           4$:     TSTB    DRVLST(R1)      ;SEE IF THIS DRIVE IS LISTED
4648  013760  001414                           BEQ     8$              ;BR IF DRIVE NOT LISTED
4649  013762  005700                           TST     R0              ;SEE IF FIRST TIME HERE
4650  013764  001402                           BEQ     6$              ;BR IF FIRST TIME HERE
4651  013766  104401  012412                   TYPE    ,COMMA          ;TYPE A COMMA
4652  013772  010137  005542           6$:     MOV     R1,SCRACH       ;USE SCRACH FOR BUFFER
4653  013776  052737  000060  005542           BIS     #'0,SCRACH      ;CONVERT DRIVE NO. TO ASCII
4654  014004  104401  005542                   TYPE    ,SCRACH         ;TYPE DRIVE NO.
4655  014010  005200                           INC     R0              ;INCREMENT NO. OF LISTED DRIVES
4656  014012  005201                   8$:     INC     R1              ;INCREMENT DRIVE NO.
4657  014014  006303                           ASL     R3              ;SHIFT BIT POINTER
4658  014016  022701  000010                   CMP     #10,R1          ;SEE IF DONE YET
4659  014022  001354                           BNE     4$              ;BR IF NOT DONE
4660  014024  005700                           TST     R0              ;SEE IF ANY DRIVES WERE LISTED
4661  014026  001016                           BNE     26$             ;BR IF YES
4662  014030  104401  011431                   TYPE    ,NOFALS         ;TYPE ''  NONE''
4663  014034  104401  010003                   TYPE    ,NODRTS         ;TYPE ''** NO DRIVES TO TEST''
4664                                    ;                               ;     'PRESS 'CONT' WHEN RDY''
4665  014040  012703  000401                   MOV     #401,R3         ;PATTERN TO MARK DRIVES
4666  014044  012701  005606                   MOV     #DRVLST,R1      ;DRIVE LIST ADDRESS
4667  014050  010321                           MOV     R3,(R1)+        ;MARK ALL DRIVES IN LIST
4668  014052  010321                           MOV     R3,(R1)+
4669  014054  010321                           MOV     R3,(R1)+
4670  014056  010311                           MOV     R3,(R1)
4671  014060  000137  044100                   JMP     HLTPRG          ;HALT PROGRAM
4672  014064  104401  001315           26$:    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
4673  014070  105737  003116                   TSTB    MDFLAG          ;SEE IF DEFAULT MODE
4674  014074  001002                           BNE     20$             ;BR IF NOT DEFAULT MODE
4675  014076  000137  015010                   JMP     LODFLS          ;GO LOAD DEFLT ITER. COUNTS
4676  014102  122737  000013  000041   20$:    CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
4677  014110  001032                           BNE     19$             ;BR IF NOT
4678  014112  105737  005606                   TSTB    DRVLST          ;SEE IF DRIVE 0 LISTED TO TEST
4679  014116  001427                           BEQ     19$             ;BR IF NOT
```

```
 4680  014120  104401  007714          24$:    TYPE    ,REPLPK          ;TYPE "IF XXDP PACK ON DRV 0, TYPE Y <CR>
 4681                                                                   ;    AND REPLACE IT"
 4682  014124  004737  030560                  JSR     PC,RDCHRS        ;READ RESPONSE
 4683  014130  014120                          24$                      ;(^C) RETURN
 4684  014132  014120                          24$                      ;(^Z) RETURN
 4685  014134  014120                          24$                      ;(^U) RETURN
 4686  014136  005700                          TST     R0               ;SEE IF NULL INPUT
 4687  014140  001416                          BEQ     19$              ;BR IF JUST <CR> TYPED
 4688  014142  022737  000131  005272          CMP     #'Y,BUFFO        ;SEE IF "Y <CR>" TYPED
 4689  014150  001363                          BNE     24$              ;BR IF NOT, TO ASK AGAIN
 4690  014152  104401  010032                  TYPE    ,CNTRDY          ;TYPE "PRESS CONT WHEN DRV RDY"
 4691  014156  042762  000100  000000          BIC     #IE,RKCS1(R2)    ;DISABLE RK06 INTERRUPT
 4692  014164  000000                          HALT                     ;HALT FOR PACK CHANGE
 4693  014166  004737  027450                  JSR     PC,INITSS        ;INIT THE SUB-SYS
 4694  014172  000137  014450                  JMP     CHKLST           ;GO CHECK STATUS OF LISTED DRIVES
 4695  014176  104401  012442          19$:    TYPE    ,PROMPT          ;TYPE ASTERISK AND SPACE
 4696                                  ;READ AND CHECK NEW DRIVE NUMBERS
 4697  014202  004737  030560                  JSR     PC,RDCHRS        ;READ IN INPUT STRING
 4698  014206  013660                          DRVTST                   ;(^C) RETURN ADDRESS
 4699  014210  013660                          DRVTST                   ;(^Z) RETURN ADDRESS
 4700  014212  013740                          TELDRV                   ;(^U) OR ERROR RETURN ADDRESS
 4701  014214  005700                          TST     R0               ;SEE IF NULL INPUT
 4702  014216  001007                          BNE     9$               ;BR IF NEW DRIVES WILL BE SELECTED
 4703  014220  105737  003136                  TSTB    DULACS           ;SEE IF DUAL-ACCESS FLAG SET
 4704  014224  001002                          BNE     22$              ;BR IF YES
 4705  014226  000137  014522                  JMP     ASKTST           ;JUMP TO THE TEST INPUT ROUTINE
 4706  014232  000137  015266          22$:    JMP     INPUTP           ;GO ASK FOR INPUT PARAMETERS
 4707  014236  022700  000017          9$:     CMP     #15.,R0          ;SEE IF TOO MANY CHARACTERS TYPED
 4708  014242  002005                          BGE     12$              ;BR IF NOT TOO MANY
 4709  014244  104401  005272          10$:    TYPE    ,BUFFO           ;ECHO BAD INPUT
 4710  014250  104401  001314                  TYPE    ,$QUES           ;TYPE <?> AND <CR>,<LF>
 4711  01    4 000631                          BR      TELDRV           ;GO TYPE DRIVES AND ASK AGAIN
 4712  014     005001                  12$:    CLR     R1               ;CLEAR CHAR. POINTER
 4713  '14261  126127  005272  000060  14$:    CMPB    BUFFO(R1),#'0    ;SEE IF DRIVE NO. < 0
 4714     26   002766                          BLT     10$              ;BR TO ECHO BAD INPUT
 4715  014270  126127  005272  000067          CMPB    BUFFO(R1),#'7    ;SEE IF > 7
 4716  014276  003362                          BGT     10$              ;BR TO ECHO BAD INPUT
 4717  014300  005201                          INC     R1               ;INCREMENT CHAR POINTER
 4718  014302  020100                          CMP     R1,R0            ;SEE IF MORE CHARS TO CHECK
 4719  014304  001406                          BEQ     16$              ;BR IF ALL CHARS CHECKED
 4720  014306  126127  005272  000054          CMPB    BUFFO(R1),#',    ;SEE IF THIS IS COMMA
 4721  014314  001353                          BNE     10$              ;BR IF NOT COMMA
 4722  014316  005201                          INC     R1               ;INCREMENT CHAR POINTER
 4723  014320  000757                          BR      14$              ;BR TO CONTINUE CHECKING CHARS
 4724                                  ;GET NEW DRIVE NUMBERS INTO LIST
 4725  014322  012701  005606          16$:    MOV     #DRVLST,R1       ;GET DRIVE LIST ADDRESS
 4726  014326  005021                          CLR     (R1)+            ;CLEAR OUT THE DRIVE LIST
 4727  014330  005021                          CLR     (R1)+
 4728  014332  005021                          CLR     (R1)+
 4729  014334  005011                          CLR     (R1)
 4730  014336  005000                          CLR     R0               ;CLEAR BUFFER POINTER
 4731  014340  116001  005272          18$:    MOVB    BUFFO(R0),R1     ;GET A DRIVE NUMBER
 4732  014344  042701  000060                  BIC     #'0,R1           ;STRIP ASCII BITS
 4733  014350  112761  000001  005606          MOVB    #1,DRVLST(R1)    ;MARK THIS DRIVE IN DRIVE LIST
 4734  014356  062700  000002                  ADD     #2,R0            ;POINT TO NEXT DRIVE NUMBER
 4735  014362  105760  005271                  TSTB    BUFFO-1(R0)      ;SEE IF NULL CHAR YET
```

```
 4736  014366  001364                        BNE     18$             ;BR IF NOT DONE YET
 4737  014370  000427                        BR      CHKLST          ;GO CHECK NEW DRIVES FOR VALID STATUS
 4738                                ;COME HERE IF ALL DRIVES REQUESTED
 4739  014372  005000                ALLDRV: CLR     R0              ;CLEAR DRIVE NUMBER
 4740  014374  013703  001376                MOV     $DEVM,R3        ;GET APT DEVICE MAP
 4741  014400  132737  000200  001341        BITB    #BIT7,$ENVM     ;SEE IF NO SIZING
 4742  014406  001002                        BNE     1$              ;BR IF NO SIZING
 4743  014410  012703  000377                MOV     #377,R3         ;SET UP FOR SIZING
 4744  014414  012701  000001        1$:     MOV     #BIT0,R1        ;SET BIT POINTER
 4745  014420  105060  005606        2$:     CLRB    DRVLST(R0)      ;INITIALIZE DRIVE ENTRY
 4746  014424  030103                        BIT     R1,R3           ;SEE IF THIS DRIVE IS REQUESTED
 4747  014426  001403                        BEQ     4$              ;BR IF NOT
 4748  014430  112760  000001  005606        MOVB    #1,DRVLST(R0)   ;MARK THIS DRIVE IN DRIVE LIST
 4749  014436  006301                4$:     ASL     R1              ;SHIFT BIT POINTER
 4750  014440  005200                        INC     R0              ;INCREMENT DRIVE NUMBER
 4751  014442  022700  000010                CMP     #10,R0          ;SEE IF DONE MARKING ALL DRIVES
 4752  014446  001364                        BNE     2$              ;BR IF NOT DONE YET
 4753                                ;CHECK ALL DRIVES MARKED IN DRIVE LIST FOR PROPER STATUS
 4754  014450  005000                CHKLST: CLR     R0              ;CLEAR DRIVE NUMBER
 4755  014452  105760  005606        2$:     TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
 4756  014456  001410                        BEQ     4$              ;BR IF NOT MARKED
 4757  014460  010037  005510                MOV     R0,DRIVE        ;SET DRIVE NUMBER FOR SCNDRV
 4758  014464  004737  027604                JSR     PC,SCNDRV       ;CHECK STATUS OF THIS DRIVE
 4759                                                                ;IF NOT VALID, TYPE MESSAGE
 4760                                                                ;   AND REMOVE IT FROM DRIVE LIST
 4761  014470  014514                        6$                      ;ERROR RETURN ADDRESS FOR SCNDRV
 4762  014472  113760  003124  005616        MOVB    TYPFMT,DTYLST(R0)       ;SET DRV TYPE. 0=RK06, 4=RK07
 4763  014500  005200                4$:     INC     R0              ;RETURN HERE IF DRIVE IS USEABLE
 4764  014502  022700  000010                CMP     #10,R0          ;SEE IF DONE CHECKING LIST
 4765  014506  001361                        BNE     2$              ;BR IF NOT DONE YET
 4766  014510  000137  013740                JMP     TELDRV          ;GO BACK TO LIST SELECTED DRIVES
 4767  014514  105060  005606        5$:     CLRB    DRVLST(R0)      ;REMOVE INVALID DRIVE FROM LIST
 4768  014520  000767                        BR      4$              ;CONTINUE CHECKING THE LIST
 4769
 4770
 4771                                ;;*****************************************************************
 4772                                .SBTTL  TO - DESIRED TEST INPUT ROUTINE
 4773                                ;*THIS ROUTINE ALLOWS INPUT OF DESIRED TESTS AND ITERATION
 4774                                ;*COUNT FOR EACH TEST. THE INPUT IS CHECKED FOR VALIDITY,
 4775                                ;*AND IF VALID, THE ITERATION COUNTS ARE LOADED INTO THE
 4776                                ;*TEST LIST (TSTLST). AN ITERATION VALUE OF 0 INDICATES
 4777                                ;*THAT THE TEST IS NOT TO BE RUN.
 4778                                ;*THIS ROUTINE REQUESTS "ENTER L,C, OR I". TYPING (L)
 4779                                ;*CAUSES THE CURRENT LIST OF TESTS AND ITERATIONS TO BE
 4780                                ;*TYPED, (C) ALLOWS TESTS AND ITERATIONS TO BE CHANGED,
 4781                                ;*AND (I) ALLOWS THE OPERATOR TO PROCEED WITH SELECTION
 4782                                ;*OF OPERATING PARAMETERS, AND RUN TESTS.
 4783                                ;*TYPING (^C) CAUSES IMMEDIATE RETURN TO DRVTST. TYPING
 4784                                ;* (^Z) CAUSES RETURN TO ASKTMD.
 4785                                ;;*****************************************************************
 4786
 4787                                ;DETERMINE L,C, OR I MODE
 4788  014522  104401  010205        ASKTST: TYPE    ,TSTMDS         ;ASK FOR TEST INPUT MODE
 4789  014526  104401  010307        ASKTMD: TYPE    ,ENTLCI         ;TYPE "ENTER L,C, OR I"
 4790  014532  004737  030560                JSR     PC,RDCHRS       ;READ RESPONSE
 4791  014536  013660                        DRVTST                  ;(^C) RETURN ADDRESS
```

```
 4792  014540  014526                        ASKTMD                  ;(^Z) RETURN ADDRESS
 4793  014542  014526                        ASKTMD                  ;(^U) OR ERROR RETURN ADDRESS
 4794  014544  005700                        TST     R0              ;SEE IF ANY INPUT
 4795  014546  001005                        BNE     4$              ;BR IF ANY INPUT
 4796  014550  104401  005272          2$:   TYPE    ,BUFF0          ;ECHO BAD INPUT
 4797  014554  104401  001314                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 4798  014560  000762                        BR      ASKTMD          ;GO ASK AGAIN
 4799  014562  022737  000114  005272  4$:   CMP     # L,BUFF0       ;SEE IF (L) TYPED
 4800  014570  001002                        BNE     6$              ;BR IF NOT (L)
 4801  014572  000137  014626                JMP     LSTTST          ;JUMP TO LIST TESTS
 4802  014576  022737  000103  005272  6$:   CMP     #'C,BUFF0       ;SEE IF (C) TYPED
 4803  014604  001002                        BNE     8$              ;BR IF NOT (C)
 4804  014606  000137  014744                JMP     CHGTST          ;JUMP TO CHANGE TESTS
 4805  014612  022737  000111  005272  8$:   CMP     #'I,BUFF0       ;SEE IF (I) TYPED
 4806  014620  001353                        BNE     2$              ;BR IF NOT (I), TO ECHO BAD INPUT
 4807  014622  000137  015266                JMP     INPUTP          ;GO INPUT PARAMS AND RUN TESTS
 4808
 4809                                  ;LIST CURRENT TESTS AND ITERATION COUNTS
 4810  014626  104401  010411          LSTTST: TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
 4811  014632  005001                        CLR     R1              ;INITIALIZE TEST INDEX
 4812  014634  004737  025766          2$:   JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 4813  014640  004737  031072                JSR     PC,TYPTST       ;TYPE CURRENT TEST AND ITERATION NUMBER
 4814  014644  104401  001315                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 4815  014650  005737  005532                TST     INTCHR          ;SEE IF ANY INPUT
 4816  014654  001416                        BEQ     8$              ;BR IF NO INPUT
 4817  014656  122737  000003  005532        CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
 4818  014664  001002                        BNE     4$              ;BR IF NOT (^C)
 4819  014666  000137  013660                JMP     DRVTST          ;JUMP TO ASK FOR DRIVES AGAIN
 4820  014672  122737  000032  005532  4$:   CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
 4821  014700  001002                        BNE     6$              ;BR IF NOT (^Z)
 4822  014702  000137  014526                JMP     ASKTMD          ;JUMP TO ASK FOR NEW TEST INPUT MODE
 4823  014706  004737  026006          6$:   JSR     PC,ECOBAD       ;ECHO BAD INPUT
 4824  014712  062701  000002          8$:   ADD     #2,R1           ;INCREMENT TEST INDEX
 4825  014716  010102                        MOV     R1,R2           ;GET COPY OF INDEX
 4826  014720  062702  005626                ADD     #TSTLST,R2      ;GET POSITION IN LIST
 4827  014724  022702  005670                CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
 4828  014730  001341                        BNE     2$              ;BR IF NOT DONE YET
 4829  014732  042777  000100  164204        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 4830  014740  000137  014526                JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
 4831
 4832                                  ;CHANGE CURRENT TESTS AND ITERATION COUNTS
 4833  014744  104401  010335          CHGTST: TYPE    ,DFTEST         ;ASK IF TESTS SHOULD BE DEFAULTED
 4834  014750  004737  030560                JSR     PC,RDCHRS       ;READ RESPONSE
 4835  014754  013660                        DRVTST                  ;(^C) RETURN ADDRESS
 4836  014756  014526                        ASKTMD                  ;(^Z) RETURN ADDRESS
 4837  014760  014744                        CHGTST                  ;(^U) OR ERROR RETURN ADDRESS
 4838  014762  005700                        TST     R0              ;SEE IF NULL INPUT
 4839  014764  001426                        BEQ     NULINP          ;BR IF NULL INPUT
 4840  014766  022737  000104  005272        CMP     #'D,BUFF0       ;SEE IF (D) TYPED
 4841  014774  001405                        BEQ     LODFLS          ;BR IF DEFAULTS REQUESTED
 4842  014776  104401  005272                TYPE    ,BUFF0          ;ECHO BAD INPUT
 4843  015002  104401  001314                TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 4844  015006  000756                        BR      CHGTST          ;GO ASK AGAIN
 4845                                  ;LOAD DEFAULT ITERATION COUNTS INTO TEST LIST
 4846  015010  012700  005670          LODFLS: MOV     #DFLTST,R0      ;LOAD DEFAULT LIST ADDRESS
 4847  015014  012702  005626                MOV     #TSTLST,R2      ;LOAD TEST LIST ADDRESS
```

```
 4848  015020  012022                  4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
 4849  015022  022702  005670                  CMP     #DFLTST,R2      ;SEE IF DONE YET
 4850  015026  001374                          BNE     4$              ;BR IF NOT DONE YET
 4851  015030  105737  003116                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE
 4852  015034  001234                          BNE     ASKTMD          ;BR IF NOT DEFAULT MODE
 4853  015036  000137  016172                  JMP     LODFPT          ;GO LOAD DEFAULT PARAMETERS
 4854  015042  005001                  NULINP: CLR     R1              ;INITIALIZE TEST INDEX
 4855  015044  104401  010411                  TYPE    ,TLSTHD         ;TYPE HEADING "TEST     ITERATIONS"
 4856                                  ;TYPE CURRENT TEST AND ITERATION NUMBER
 4857  015050  004737  031072          8$:     JSR     PC,TYPTST       ;TYPE A TEST NO. AND ITERATION NO.
 4858  015054  104401  012421                  TYPE    ,SPACE1         ;TYPE A SPACE
 4859  015060  104401  012442                  TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
 4860                                  ;READ AND CHECK INPUT, IF ANY
 4861  015064  004737  030560                  JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
 4862  015070  013660                          DRVTST                  ;(^C) RETURN ADDRESS
 4863  015072  014526                          ASKTMD                  ;(^Z) RETURN ADDRESS
 4864  015074  015050                          8$                      ;(^U) OR ERROR RETURN ADDRESS
 4865  015076  005700                          TST     R0              ;SEE IF ANY INPUT
 4866  015100  001012                          BNE     12$             ;BR IF ANY INPUT
 4867  015102  062701  000002          10$:    ADD     #2,R1           ;INCREMENT INDEX
 4868  015106  010102                          MOV     R1,R2           ;COPY INDEX
 4869  015110  062702  005626                  ADD     #TSTLST,R2      ;GET POSITION IN LIST
 4870  015114  022702  005670                  CMP     #DFLTST,R2      ;SEE IF DONE WITH LIST
 4871  015120  001353                          BNE     8$              ;BR IF NOT DONE YET
 4872  015122  000137  014526                  JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
 4873  015126  022737  000041  005272  12$:    CMP     #'!,BUFFO       ;SEE IF (!) TYPED
 4874  015134  001431                          BEQ     16$             ;BR TO PROPAGATE CURRENT ITER. NO.
 4875  015136  005002                          CLR     R2              ;INITIALIZE (!) INDICATOR
 4876  015140  122760  000041  005271          CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (!)
 4877  015146  001004                          BNE     14$             ;BR IF NOT (!)
 4878  015150  105060  005271                  CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
 4879  015154  005300                          DEC     R0              ;DECREMENT CHAR COUNT
 4880  015156  005202                          INC     R2              ;SET (!) INDICATOR
 4881  015160  022700  000005          14$:    CMP     #5,R0           ;SEE HOW MANY CHARS NOW
 4882  015164  002433                          BLT     20$             ;BR IF TOO MANY (MAX ITER. NO. = 77776)
 4883  015166  012746  005272                  MOV     #BUFFO,-(SP)    ;GET BUF ADDR. ON STACK FOR OCTBIN
 4884  015172  004737  052430                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
 4885  015176  015254                          20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
 4886  015200  012600                          MOV     (SP)+,R0        ;GET ITERATION NUMBER
 4887  015202  020027  077776                  CMP     R0,#77776       ;SEE IF > 77776
 4888  015206  101022                          BHI     20$             ;BR IF TOO BIG
 4889  015210  010061  005626                  MOV     R0,TSTLST(R1)   ;PUT ITERATION NUMBER INTO TEST LIST
 4890  015214  005702                          TST     R2              ;SEE IF (!) WAS TYPED
 4891  015216  001731                          BEQ     10$             ;BR IF NOT (!)
 4892                                  ;PROPAGATE CURRENT ITERATION NO. TO THE END OF TSTLST
 4893  015220  010102                  16$:    MOV     R1,R2           ;COPY INDEX
 4894  015222  062702  005626                  ADD     #TSTLST,R2      ;GET POSITION IN LIST
 4895  015226  022702  005666                  CMP     #DFLTST-2,R2    ;SEE IF AT END OF LIST
 4896  015232  001002                          BNE     17$             ;BR IF NOT DONE
 4897  015234  000137  014526                  JMP     ASKTMD          ;GO ASK FOR NEW TEST INPUT MODE
 4898  015240  016161  005626  005630  17$:    MOV     TSTLST(R1),TSTLST+2(R1) ;PROPAGATE TO NEXT WORD
 4899  015246  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
 4900  015252  000762                          BR      16$             ;BR TO CONTINUE
 4901  015254  104401  005272          20$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
 4902  015260  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 4903  015264  000671                          BR      8$              ;GO ASK AGAIN
```

```
4904
4905                                    ;ASK FOR DESIRED OPERATIONAL PARAMETER INPUT MODE
4906  015266  104401  010441            INPUTP: TYPE    ,EXPLAN         ;EXPLAIN INPUT MODES
4907  015272  005037  005512            ASKMDE: CLR     STALLS          ;INHIBIT STALL BETWEEN OPERATIONS
4908  015276  104401  010567                    TYPE    ,PARMDE         ;ASK FOR DESIRED INPUT MODE
4909  015302  104401  012442                    TYPE    ,PROMPT         ;TYPE "* "
4910  015306  004737  030560                    JSR     PC,RDCHRS       ;READ RESPONSE TO MODE QUESTION
4911  015312  013660                            DRVTST                  ;(^C) RETURN ADDRESS
4912  015314  015272                            ASKMDE                  ;(^Z) RETURN ADDRESS
4913  015316  015272                            ASKMDE                  ;(^U) OR ERROR RETURN ADDRESS
4914  015320  005700                            TST     R0              ;SEE IF NULL INPUT
4915  015322  001005                            BNE     4$              ;BR IF ANY INPUT
4916  015324  104401  005272            2$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
4917  015330  104401  001314                    TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
4918  015334  000756                            BR      ASKMDE          ;GO ASK AGAIN
4919  015336  022737  000124  005272  4$:     CMP     #'T,BUFFO       ;SEE IF (T) TYPED
4920  015344  001002                            BNE     6$              ;BR IF NOT (T)
4921  015346  000137  015654                    JMP     TYPLST          ;JUMP TO THE TYPE LIST ROUTINE
4922  015352  022737  000117  005272  6$:     CMP     #'O,BUFFO       ;SEE IF (O) TYPED
4923  015360  001002                            BNE     8$              ;BR IF NOT (O)
4924  015362  000137  016126                    JMP     OPNLST          ;JUMP TO THE OPEN LIST ROUTINE
4925  015366  022737  000123  005272  8$:     CMP     #'S,BUFFO       ;SEE IF (S) TYPED
4926  015374  001002                            BNE     10$             ;BR IF NOT (S)
4927  015376  000137  016412                    JMP     SETPRM          ;JUMP TO THE SET INDIV. PARAM. ROUTINE
4928  015402  022737  000122  005272  10$:    CMP     #'R,BUFFO       ;SEE IF (R) TYPED
4929  015410  001345                            BNE     2$              ;BR IF NOT (R), TO ECHO BAD INPUT
4930  015412  023737  005746  005750            CMP     S0,S1           ;COMPARE 20(DEC) SECTOR LIMITS
4931  015420  003403                            BLE     12$             ;BR IF S0 NOT > S1
4932  015422  104401  010670                    TYPE    ,SECNL1         ;TYPE "S0>S1 NOT ALLOWED"
4933  015426  000721                            BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
4934  015430  023737  005752  005754  12$:    CMP     S2,S3           ;COMPARE 22(DEC) SECTOR LIMITS
4935  015436  003405                            BLE     14$             ;BR IF S2 NOT > S3
4936  015440  104401  010720                    TYPE    ,SECNL2         ;TYPE "S2>S3"
4937  015444  104401  010701                    TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
4938  015450  000710                            BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
4939  015452  023737  005740  005742  14$:    CMP     FT,LT           ;COMPARE CHOSEN TRACK LIMITS
4940  015460  003405                            BLE     20$             ;BR IF FT NOT > LT
4941  015462  104401  010732                    TYPE    ,TRKNLW         ;TYPE "FT>LT"
4942  015466  104401  010701                    TYPE    ,NOTALD         ;TYPE "NOT ALLOWED"
4943  015472  000677                            BR      ASKMDE          ;GO ASK AGAIN FOR PARAMETERS
4944                                    ;CHECK WORD COUNT AGAINST PHYSICAL MEMORY AVAILABLE
4945  015474  013700  005766            20$:    MOV     WC,R0           ;WORD COUNT LO BITS
4946  015500  005300                            DEC     R0              ;DECREMENT BY 1
4947  015502  005001                            CLR     R1              ;WORD COUNT HI BITS
4948  015504  000241                            CLC                     ;CONVERT WORD COUNT TO BYTES
4949  015506  006100                            ROL     R0
495C  015510  006101                            ROL     R1
4951  015512  063700  005762                    ADD     MA,R0           ;ADD WORD COUNT (WC) TO MEM. ADR. (MA)
4952  015516  005501                            ADC     R1
4953  015520  063701  005764                    ADD     MA+2,R1
4954  015524  020137  006130                    CMP     R1,MAHILM+2     ;COMPARE HI BITS
4955  015530  101004                            BHI     26$             ;BR IF WC TOO BIG
4956  015532  001006                            BNE     28$             ;BR IF WC IS OK
4957  015534  020037  006126                    CMP     R0,MAHILM       ;COMPARE LO BITS
4958  015540  101403                            BLOS    28$             ;BR IF WC IS OK
4959  015542  104401  010744            26$:    TYPE    ,WC2BIG         ;TYPE "**  WC OR MA TOO LARGE"
```

```
 4960  015546  000651                          BR      ASKMDE          ;GO BACK TO ASK AGAIN FOR PARAMS
 4961                              ;SAVE XXDP LOADER, IF NECESSARY
 4962  015550  004737  026726      28$:    JSR     PC,CHKXDP       ;GO SAVE XXDP LOADER IF NECESSARY
 4963  015554  015562                      36$                     ;"NO SAVE" RETURN ADDRESS
 4964  015556  000137  016646      30$:    JMP     RUNTST          ;JUMP TO THE RUN TESTS ROUTINE
 4965  015562  105737  000041      36$:    TSTB    @#41            ;SEE IF LOADED BY XXDP
 4966  015566  001773                      BEQ     30$             ;BR IF NOT
 4967  015570  104401  007404              TYPE    ,OVLODR         ;TYPE "OVERLAY LOADER ? (Y OR N) *"
 4968  015574  004737  030560              JSR     PC,RDCHRS       ;READ RESPONSE
 4969  015600  015562                      36$                     ;(^C) RETURN ADDRESS
 4970  015602  015562                      36$                     ;(^Z) RETURN ADDRESS
 4971  015604  015562                      36$                     ;(^U) OR ERROR RETURN ADDRESS
 4972  015606  022700  000001              CMP     #1,R0           ;SEE IF 1 CHAR TYPED
 4973  015612  001405                      BEQ     38$             ;BR IF 1 CHAR TYPED
 4974  015614  104401  005272      37$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
 4975  015620  104401  001314              TYPE    ,$QUES
 4976  015624  000756                      BR      36$             ;GO ASK AGAIN
 4977  015626  122737  000131  005272  38$:    CMPB    #'Y,BUFFO       ;SEE IF (Y) TYPED
 4978  015634  001001                      BNE     40$             ;BR IF NOT (Y)
 4979  015636  000747                      BR      30$             ;GO RUN TESTS
 4980  015640  122737  000116  005272  40$:    CMPB    #'N,BUFFO       ;SEE IF (N) TYPED
 4981  015646  001362                      BNE     37$             ;BR IF NOT (N)
 4982  015650  000137  015272              JMP     ASKMDE          ;GO ASK FOR PARAMETERS AGAIN
 4983
 4984
 4985
 4986                              .SBTTL  TO - TYPE (T) LIST ROUTINE
 4987                              ;*THIS ROUTINE TYPES THE ENTIRE PARAMETER LIST ON THE
 4988                              ;*CONSOLE, LINE BY LINE. EACH LINE HAS THE FOLLOWING
 4989                              ;*FORMAT:  XX=YYYYYY , WHERE XX IS A PARAMETER MNEMONIC
 4990                              ;*AND YYYYYY IS THE VALUE OF THE PARAMETER IN OCTAL.
 4991                              ;*TYPING (^C) CAUSES IMMEDIATE RETURN TO DRVTST,
 4992                              ;*AND (^Z) CAUSES RETURN TO ASKMDE.
 4993
 4994  015654                      TYPLST:
 4995  015654  005001                      CLR     R1              ;INITIALIZE INDEX
 4996  015656  004737  025766              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 4997  015662  004737  031126      1$:     JSR     PC,TYPPRM       ;TYPE CURRENT PARAMETER AND VALUE
 4998  015666  104401  001315              TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 4999  015672  005737  005532              TST     INTCHR          ;SEE IF ANY INPUT AT KBD
 5000  015676  001420                      BEQ     8$              ;BR IF NO INPUT
 5001  015700  122737  000003  005532      CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
 5002  015706  001002                      BNE     4$              ;BR IF NOT (^C)
 5003  015710  000137  013660              JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
 5004  015714  122737  000032  005532  4$:     CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
 5005  015722  001002                      BNE     6$              ;BR IF NOT (^Z)
 5006  015724  000137  015272              JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5007  015730  004737  026006      6$:     JSR     PC,ECOBAD       ;ECHO BAD INPUT
 5008  015734  004737  025766              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 5009  015740  022761  040515  006152  8$:     CMP     #'MA,PRMNEM(R1) ;SEE IF PARAM. IS (MA)
 5010  015746  001002                      BNE     10$             ;BR IF NOT (MA)
 5011  015750  062701  000002              ADD     #2,R1           ;INCREMENT PARAMETER INDEX
 5012  015754  062701  000002      10$:    ADD     #2,R1           ;INCREMENT PARAMETER INDEX
 5013  015760  010102                      MOV     R1,R2           ;GET COPY OF INDEX
 5014  015762 *062702  005732              ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
 5015  015766  022702  005776              CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
```

```
 5016  015772  001333                        BNE     1$              ;BR IF NOT DONE YET
 5017  015774  032737  100000  005760        BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
 5018  016002  001005                        BNE     12$             ;BR IF PATTERN SPECIFIED
 5019  016004  042777  000100  163132        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 5020  016012  000137  015272                JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5021                                ;TYPE OUT USER-DEFINED PATTERN 15
 5022  016016  104401  011057        12$:    TYPE    ,PFIFTN         ;TYPE 'USER-DEFINED PATTERN 15 :"
 5023  016022  005001                        CLR     R1              ;INITIALIZE WORD INDEX
 5024  016024  005737  005532        14$:    TST     INTCHR          ;SEE IF ANY INPUT
 5025  016030  001420                        BEQ     20$             ;BR IF NO INPUT
 5026  016032  122737  000003  005532        CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
 5027  016040  001002                        BNE     16$             ;BR IF NOT (^C)
 5028  016042  000137  013660                JMP     DRVTST          ;JUMP TO ASK FOR DRIVE(S) AGAIN
 5029  016046  122737  000032  005532  16$:  CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
 5030  016054  001002                        BNE     18$             ;BR IF NOT (^Z)
 5031  016056  000137  015272                JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5032  016062  004737  026006        18$:    JSR     PC,ECOBAD       ;ECHO BAD INPUT
 5033  016066  004737  025766                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 5034  016072  004737  031206        20$:    JSR     PC,TYPPAT       ;TYPE WORD XX = YYYYYY
 5035  016076  104401  001315                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 5036  016102  062701  000002                ADD     #2,R1           ;INCREMENT INDEX
 5037  016106  022701  000040                CMP     #32.,R1         ;SEE IF 16 WORDS TYPED
 5038  016112  001344                        BNE     14$             ;BR IF NOT DONE YET
 5039  016114  042777  000100  163022        BIC     #BIT6,@$TKS     ;DISABLE KBD INTERRUPT
 5040  016122  000137  015272                JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
 5041
 5042
 5043
 5044                                .SBTTL  TO - OPEN (O) LIST ROUTINE
 5045                                ;*THIS ROUTINE IS USED TO EITHER REQUEST LOADING OF ALL
 5046                                ;*DEFAULT PARAMETERS INTO THE PARAMETER LIST, OR TO
 5047                                ;*SEQUENTIALLY OPEN EACH PARAMETER FOR ALTERATION BY
 5048                                ;*TTY INPUT. TYPING (^C) CAUSES IMMEDIATE RETURN TO
 5049                                ;*DRVTST, AND (^Z) CAUSES RETURN TO ASKMDE.
 5050
 5051
 5052  016126                        OPNLST:
 5053  016126  104401  010775                TYPE    ,DFQUES         ;ASK IF ALL DEFAULT VALUES DESIRED
 5054  016132  004737  030560                JSR     PC,RDCHRS       ;READ RESPONSE TO DEFAULT QUESTION
 5055  016136  013660                        DRVTST                  ;(^C) RETURN ADDRESS
 5056  016140  015272                        ASKMDE                  ;(^Z) RETURN ADDRESS
 5057  016142  016126                        OPNLST                  ;(^U) OR ERROR RETURN ADDRESS
 5058  016144  005700                        TST     R0              ;SEE IF NULL INPUT
 5059  016146  001433                        BEQ     NOTDFT          ;BR IF DEFAULTS NOT REQUESTED
 5060  016150  022737  000104  005272        CMP     #'D,BUFFO       ;SEE IF (D) TYPED
 5061  016156  001405                        BEQ     LODFPT          ;BR IF DEFAULTS DESIRED
 5062  016160  104401  005272                TYPE    ,BUFFO          ;ECHO BAD INPUT
 5063  016164  104401  001314                TYPE    ,$QUES
 5064  016170  000756                        BR      OPNLST          ;GO ASK AGAIN
 5065                                ;LOAD ALL DEFAULT VALUES INTO PARAMETER LIST
 5066  016172  012700  005776        LODFPT: MOV     #PRDFLT,R0      ;LOAD DEFAULT LIST ADDRESS
 5067  016176  012702  005732                MOV     #PRMLST,R2      ;LOAD PARAMETER LIST ADDRESS
 5068  016202  012022                4$:     MOV     (R0)+,(R2)+     ;LOAD A DEFAULT VALUE
 5069  016204  022702  005776                CMP     #PRDFLT,R2      ;SEE IF DONE YET
 5070  016210  001374                        BNE     4$              ;BR IF MORE VALUES TO LOAD YET
 5071  016212  105737  003116                TSTB    MDFLAG          ;SEE IF DEFAULT MODE
```

```
                        BNE     6$              ;BR IF NOT DEFAULT MODE
                        MOV     #1,$EOPCT       ;SET PASS COUNT = 1
                        JMP     STPASS          ;GO RUN DFLT TESTS
                6$:     JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
                NOTDFT: CLR     R1              ;INITIALIZE INDEX
                ;TYPE CURRENT PARAMETER AND VALUE
                8$:     JSR     PC,TYPPRM       ;TYPE PARAMETER AND VALUE
                        TYPE    ,SPACE1         ;TYPE A SPACE
                        TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
                ;READ AND CHECK INPUT, IF ANY
                        JSR     PC,RDCHRS       ;READ OCTAL DIGITS TYPED
                        DRVTST                  ;(^C) RETURN ADDRESS FOR RDCHRS
                        ASKMDE                  ;(^Z) RETURN ADDRESS FOR RDCHRS
                        8$                      ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
                        TST     R0              ;SEE IF ANY INPUT
                        BNE     10$             ;BR IF ANY INPUT
                        CMP     #''MA,PRMNEM(R1) ;SEE IF (MA) JUST DEFAULTED
                        BNE     12$             ;BR IF NOT (MA)
                        ADD     #2,R1           ;INCREMENT INDEX
                        BR      12$             ;BR TO MOVE ON TO NEXT PARAMETER
                10$:    CMP     #10,R0          ;SEE IF 8 CHARACTERS TYPED
                        BLT     14$             ;BR IF MORE THAN 8 TYPED
                        MOV     #BUFFO,-(SP)    ;PUT BUFFER ADDR. ON STACK FOR OCTBIN
                        JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
                        14$                     ;ERROR RETURN ADDRESS FOR OCTBIN
                        MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
                        MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
                ;CHECK PARAMETER VALUE AND PUT INTO LIST
                        JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAM VALUE
                        14$                     ;ERROR RETURN ADDR. FOR CHKPRM
                ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
                12$:    JSR     PC,MODP15
                ;MOVE ON TO NEXT PARAMETER
                        ADD     #2,R1           ;INCREMENT THE PARAMETER INDEX
                        MOV     R1,R2           ;GET COPY OF INDEX
                        ADD     #PRMLST,R2      ;COMPUTE POSITION IN LIST
                        CMP     #PRDFLT,R2      ;SEE IF DONE WITH LIST
                        BNE     8$              ;BR IF NOT DONE YET
                        JMP     ASKMDE          ;JUMP TO ASK FOR NEW MODE
                14$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
                        TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
                        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.
```

```
5128  016412                          SETPRM:
5129  016412  104401  012445                  TYPE    ,PRMPSP         ;TYPE '> '' PROMPTER
5130                                  ;READ AND CHECK INPUT, IF ANY
5131  016416  004737  030560                  JSR     PC,RDCHRS       ;READ INPUT LINE
5132  016422  013660                          DRVTST                  ;(^C) RETURN ADDRESS FOR RDCHRS
5133  016424  015272                          ASKMDE                  ;(^Z) RETURN ADDRESS FOR RDCHRS
5134  016426  016412                          SETPRM                  ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
5135  016430  020027  000004                  CMP     R0,#4           ;SEE IF AT LEAST 4 CHARACTERS TYPED
5136  016434  002005                          BGE     6$              ;BR IF AT LEAST 4 TYPED
5137  016436  104401  005272          4$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
5138  016442  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
5139  016446  000761                          BR      SETPRM          ;BR TO ASK FOR INPUT AGAIN
5140  016450  020027  000013          6$:     CMP     R0,#11.         ;SEE IF ELEVEN OR LESS CHARS TYPED
5141  016454  003370                          BGT     4$              ;BR IF MORE THAN ELEVEN TYPED
5142                                  ;SEE IF THIS MNEMONIC RESIDES IN PARAM MNEMONIC TABLE
5143  016456  005001                          CLR     R1              ;INITIALIZE PARAMETER INDEX
5144  016460  023761  005272  006152  8$:     CMP     BUFFO,PRMNEM(R1) ;TRY TO MATCH 2-CHAR MNEMONIC
5145  016466  001411                          BEQ     10$             ;BR IF PARAMETER FOUND IN TABLE
5146  016470  062701  000002                  ADD     #2,R1           ;INCREMENT INDEX
5147  016474  010102                          MOV     R1,R2           ;GET COPY OF INDEX
5148  016476  062702  005732                  ADD     #PRMLST,R2      ;SEE IF ALL ENTRIES HAVE BEEN CHECKED
5149  016502  022702  005776                  CMP     #PRDFLT,R2
5150  016506  001364                          BNE     8$              ;BR IF MORE TO CHECK YET
5151  016510  000752                          BR      4$              ;BR TO ECHO BAD INPUT
5152                                  ;CHECK FOR VALID LINE FORMAT
5153  016512  012702  000002          10$:    MOV     #2,R2           ;INITIALIZE BUFFER POINTER
5154  016516  122762  000075  005272          CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS ''=''
5155  016524  001411                          BEQ     12$             ;BR IF IT IS ''=''
5156  016526  122762  000040  005272          CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHAR IS A SPACE
5157  016534  001340                          BNE     4$              ;BR TO ECHO BAD INPUT
5158  016536  005202                          INC     R2              ;INCREMENT BUFFER POINTER
5159  016540  122762  000075  005272          CMPB    #'=,BUFFO(R2)   ;SEE IF NEXT CHAR IS ''=''
5160  016546  001333                          BNE     4$              ;BR TO ECHO INVALID INPUT
5161  016550  005202                  12$:    INC     R2              ;INCREMENT BUFFER POINTER
5162  016552  020200                          CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
5163  016554  002330                          BGE     4$              ;BR TO ECHO INVALID INPUT
5164  016556  122762  000040  005272          CMPB    #040,BUFFO(R2)  ;SEE IF NEXT CHARACTER IS SPACE
5165  016564  001003                          BNE     14$             ;BR IF NOT A SPACE
5166  016566  005202                          INC     R2              ;INCREMENT BUFFER POINTER
5167  016570  020200                          CMP     R2,R0           ;SEE IF MORE CHARS LEFT IN BUFFER
5168  016572  002321                          BGE     4$              ;BR IF NONE LEFT
5169  016574  160200                  14$:    SUB     R2,R0           ;SUBTRACT POINTER FROM CHAR COUNT
5170  016576  020027  000010                  CMP     R0,#10          ;SEE HOW MANY CHARS LEFT
5171  016602  003315                          BGT     4$              ;BR IF TOO MANY LEFT
5172                                  ;CHECK FOR LEGAL PARAMETER VALUE
5173  016604  062702  005272                  ADD     #BUFFO,R2       ;GET ADDRESS OF DIGITS
5174  016610  010246                          MOV     R2,-(SP)        ;PUT IT ON STACK FOR OCTBIN
5175  016612  004737  052430                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
5176  016616  016436                          4$                      ;ERROR RETURN ADDR. FOR OCTBIN
5177  016620  012637  005476                  MOV     (SP)+,LOWOCT    ;GET LOW BINARY BITS
5178  016624  013737  052562  005500          MOV     $HIOCT,HIGOCT   ;GET HIGH BINARY BITS
5179  016632  004737  031564                  JSR     PC,CHKPRM       ;CHECK VALIDITY OF PARAMETER VALUE
5180  016636  016436                          4$                      ;ERROR RETURN ADDR. FOR CHKPRM
5181                                  ;OPEN USER-DEFINED PATTERN 15 FOR MODIFICATION, IF NECESSARY
5182  016640  004737  031240                  JSR     PC,MODP15
5183  016644  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.

016646                                  RUNTST:
                                        ;INPUT THE DESIRED NUMBER OF PROGRAM PASSES
016646  104401  011203          4$:     TYPE    ,ENTPAS         ;ASK FOR NUMBER OF PASSES
016652  004737  030560                  JSR     PC,RDCHRS       ;READ RESPONSE
016656  013660                          DRVTST                  ;(^C) RETURN ADDRESS
016660  015272                          ASKMDE                  ;(^Z) RETURN ADDRESS
016662  016646                          4$                      ;(^U) OR ERROR RETURN ADDRESS
016664  005700                          TST     R0              ;SEE IF NULL INPUT
016666  001403                          BEQ     6$              ;GO ASK AGAIN
016670  022700  000005                  CMP     #5,R0           ;SEE HOW MANY CHARS TYPED
016674  002005                          BGE     8$              ;BR IF 5 OR LESS
016676  104401  005272          6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
016702  104401  001314                  TYPE    ,$QUES
016706  000757                          BR      4$              ;GO ASK AGAIN
016710  012746  005272          8$:     MOV     #BUFFO,-(SP)    ;GET BUF ADDR ON STACK FOR OCTBIN
016714  004737  052430                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
016720  016676                          6$                      ;ERROR RETURN ADDRESS FOR OCTBIN
016722  012637  025422                  MOV     (SP)+,$EOPCT    ;SET DESIRED NO. OF PASSES (1-77777)
016726  001763                          BEQ     6$              ;BR IF 0, TO ASK AGAIN
                                        ;THIS IS THE ACTUAL START OF A RUN WITH X PASSES
016730  005037  001326          STPASS: CLR     $PASS           ;INIT. THE PASS NUMBER
016734  112737  000001  003120          MOVB    #1,ISTING       ;SET "RUNNING TESTS" FLAG
016742  012737  177777  005510  NEWPAS: MOV     #177777,DRIVE   ;INITIALIZE DRIVE NO. TO -1
016750  005037  001330                  CLR     $DEVCT          ;INIT APT DEVICE COUNT TO 0
                                        ;CHECK DRIVE LIST FOR MORE DRIVES TO TEST ON THIS PASS
016754  012737  000000  177776  NEWDRV: MOV     #PRO,@#PS       ;RE-ESTABLISH PRIORITY 0
016762  012706  001100                  MOV     #STACK,SP       ;RESTORE THE STACK
016766  105037  003130                  CLRB    DRVERS          ;CLEAR ERROR COUNT FOR CURRENT DRIVE
016772  005037  005532                  CLR     INTCHR          ;CLEAR TTY INPUT BUFFER WORD
016776  005237  005510                  INC     DRIVE           ;INCREMENT DRIVE NUMBER
017002  022737  000010  005510          CMP     #10,DRIVE       ;SEE IF DONE WITH THIS PASS
017010  001002                          BNE     2$              ;BR IF NOT DONE CHECKING LIST
017012  000137  025370                  JMP     DUNPAS          ;JUMP IF DONE WITH THIS PASS
017016  013700  005510          2$:     MOV     DRIVE,R0        ;GET NEW DRIVE NUMBER
017022  105760  005606                  TSTB    DRVLST(R0)      ;SEE IF THIS DRIVE IS MARKED IN LIST
017026  001752                          BEQ     NEWDRV          ;BR IF NOT MARKED, TO CHECK ANOTHER
017030  105037  001102                  CLRB    $TSTNM          ;INIT TEST NUMBER TO 0
017034  005037  001304                  CLR     $TIMES          ;INIT ITERATION COUNT
017040  105760  005616                  TSTB    DTYLST(R0)      ;SEE IF RK07
017044  001077                          BNE     1$              ;BR IF YES
```

```
 5240  017046  105037  003124                  CLRB    TYPFMT          ;ELSE LOAD ALL RK06 PARAMETERS
 5241  017052  012737  000625  017702          MOV     #625,LCM5
 5242  017060  012737  000631  017704          MOV     #631,LCM1
 5243  017066  012737  000631  006044          MOV     #631,PRMLIM+2
 5244  017074  012737  000632  005734          MOV     #632,LC
 5245  017102  012737  000632  017706          MOV     #632,LSTCYL
 5246  017110  012737  000632  006000          MOV     #632,PRDFLT+2
 5247  017116  012737  000632  006050          MOV     #632,PRMLIM+6
 5248  017124  012737  000632  006054          MOV     #632,PRMLIM+12
 5249  017132  012737  000633  017710          MOV     #633,LCP1
 5250  017140  012737  000400  017712          MOV     #400,HOLD1
 5251  017146  012737  001000  017714          MOV     #1000,HOLD2
 5252  017154  012737  000025  017716          MOV     #25,HOLD3
 5253  017162  012737  010025  017720          MOV     #10025,HOLD4
 5254  017170  012737  010125  017722          MOV     #10125,HOLD5
 5255  017176  012737  000002  017724          MOV     #2,HOLD6
 5256  017204  012737  110244  017726          MOV     #110244,HOLD7
 5257  017212  012737  017500  007216          MOV     #8000.,MAXLO2   ;8000 US FOR RK06
 5258  017220  012737  000000  007220          MOV     #0,MAXHI2       ;
 5259  017226  012737  022370  007222          MOV     #9464.,MAXLO4   ;75000 US FOR RK06
 5260  017234  012737  000001  007224          MOV     #1,MAXHI4       ;
 5261  017242  000477                          BR      3$
 5262
 5263  017244  152737  000004  003124  1$:     BISB    #B.CDT,TYPFMT   ;LOAD ALL RK07 PARAMETERS
 5264  017252  012737  001451  017702          MOV     #1451,LCM5
 5265  017260  012737  001455  017704          MOV     #1455,LCM1
 5266  017266  012737  001455  006044          MOV     #1455,PRMLIM+2
 5267  017274  012737  001456  005734          MOV     #1456,LC
 5268  017302  012737  001456  017706          MOV     #1456,LSTCYL
 5269  017310  012737  001456  006000          MOV     #1456,PRDFLT+2
 5270  017316  012737  001456  006050          MOV     #1456,PRMLIM+6
 5271  017324  012737  001456  006054          MOV     #1456,PRMLIM+12
 5272  017332  012737  001457  017710          MOV     #1457,LCP1
 5273  017340  012737  001000  017712          MOV     #1000,HOLD1
 5274  017346  012737  002000  017714          MOV     #2000,HOLD2
 5275  017354  012737  002025  017716          MOV     #2025,HOLD3
 5276  017362  012737  012025  017720          MOV     #12025,HOLD4
 5277  017370  012737  012125  017722          MOV     #12125,HOLD5
 5278  017376  012737  000012  017724          MOV     #12,HOLD6
 5279  017404  012737  016104  017726          MOV     #16104,HOLD7
 5280
 5281  017412  012737  014544  007216          MOV     #6500.,MAXLO2   ;6500 US FOR RK07
 5282  017420  012737  000000  007220          MOV     #0,MAXHI2       ;
 5283  017426  012737  016450  007222          MOV     #7464.,MAXLO4   ;73000 US FOR RK07
 5284  017434  012737  000001  007224          MOV     #1,MAXHI4       ;
 5285  017442  010037  001332          3$:     MOV     R0,$UNIT        ;SET DRIVE NO FOR APT
 5286                                ;READ THE HEADER ON SECTOR 0,TRACK 0,CYL 0, AND GET THE DRIVE FORMAT
 5287  017446  004737  027450                  JSR     PC,INITSS       ;INIT. DRIVER PARAMS AND S.S.
 5288  017452  112765  000125  000001          MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
 5289  017460  004737  040252                  JSR     PC,DRVCAL       ;DO READ HEADER
 5290  017464  013712  017716                  MOV     HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT - 0
 5291  017470  032712  000200          4$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
 5292  017474  001775                          BEQ     4$              ;BR IF NOT RDY YET
 5293  017476  013712  017720                  MOV     HOLD4,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
 5294  017502  032712  000200          6$:     BIT     #BIT7,(R2)      ;CHEC  FOR C. READY IN CS1
 5295  017506  001775                          BEQ     6$              ;BR IF NOT RDY YET
```

```
5296  017510  105037  003125                CLRB    FORMAT          ;INITIALIZE FORMAT BYTE TO 0
5297  017514  013737  005752  005516        MOV     S2,FS           ;INIT SECTOR LIMITS FOR 22(DEC) SECTOR FMT
5298  017522  013737  005754  005520        MOV     S3,LS
5299  017530  005762  000024                TST     RKDB(R2)        ;POP SILO ONE TIME
5300  017534  032762  001000  000024        BIT     #BIT9,RKD9(R2)  ;TEST FORMAT BIT IN HEADER WORD 2
5301  017542  001411                        BEQ     8$              ;BR IF 22 SECTOR FORMAT
5302  017544  152737  000020  003125        BISB    #B.CFMT,FORMAT  ;SET 20 SECTOR FORMAT FOR THIS DRIVE
5303  017552  013737  005746  005516        MOV     S0,FS           ;INIT SECTOR LIMITS FOR 20(DEC) SECTOR FMT
5304  017560  013737  005750  005520        MOV     S1,LS
5305  017566  013737  005772  005512  8$:   MOV     ST,STALLS       ;SET NUMBER OF UNIT STALLS DESIRED
5306  017574  004737  027450                JSR     PC,INITSS       ;INIT THE S.S.
5307  017600  004737  036630                JSR     PC,REDBSF       ;READ BAD SECTOR FILE FOR THIS DRIVE
5308  017604  113737  005510  011331        MOVB    DRIVE,DRVNO     ;GET DRIVE NO.
5309  017612  152737  000060  011331        BISB    #'0,DRVNO       ;CONVERT TO ASCII
5310  017620  104401  011312                TYPE    ,TSTDRN         ;TYPE "TESTING DRIVE X"
5311  017624  005737  001326                TST     $PASS           ;SEE IF THIS IS FIRST PASS
5312  017630  001012                        BNE     10$             ;BR IF NOT FIRST PASS
5313  017632  004737  032070                JSR     PC,DRVSER       ;TYPE "DRIVE SER. NO.  XXX"
5314  017636  004737  032210                JSR     PC,CRTSER       ;TYPE "CART. SER. NO.  XXXXXXXXXXX"
5315  017642  032737  000020  005770        BIT     #BIT4,CS        ;SEE IF BAD SECTORS SHOULD BE TYPED
5316  017650  001402                        BEQ     10$             ;BR IF NOT
5317  017652  004737  037120                JSR     PC,TYPBSF       ;TYPE BAD SECTOR FILES
5318  017656  005037  005532          10$:  CLR     INTCHR          ;INIT. TTY INPUT CHAR BUFFER
5319  017662  105737  003136                TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
5320  017666  001420                        BEQ     TST1            ;BR IF NOT
5321  017670  112737  000021  001102        MOVB    #21,$TSTNM      ;SET TEST NO.
5322  017676  000137  024444                JMP     RWDTST          ;JUMP TO R/W DATA TEST
5323
5324  017702  000000          LCM5:   0               ;LAST CYL-5
5325  017704  000000          LCM1:   0               ;LAST CYL-1
5326  017706  000000          LSTCYL: 0               ;LAST CYL 632 FOR RK06, 1456 FOR RK07
5327  017710  000000          LCP1:   0               ;LAST CYL+1
5328  017712  000000          HOLD1:  0
5329  017714  000000          HOLD2:  0
5330  017716  000000          HOLD3:  0
5331  017720  000000          HOLD4:  0
5332  017722  000000          HOLD5:  0
5333  017724  000000          HOLD6:  0
5334  017726  000000          HOLD7:  0
5335
5336
5337
5338
```

```
                                        ;;*****************************************************************
                                        ;*TEST 1        RECALIBRATE/SEEK TEST
                                        ;*
                                        ;*      THIS TEST PERFORMS A RECALIBRATE, FOLLOWED BY A
                                        ;*      SEEK TO CYLINDER LC.
                                        ;*
                                        ;;*****************************************************************
017730  000004                          TST1:   SCOPE
017732  012737  000001  001324                  MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
017740  004737  030416                          JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
017744  004737  030464                          JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
017750  000137  020010                          JMP     TST2            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
017754  004737  027450                          JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
017760  004737  025766                          JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;RECALIBRATE THE DRIVE
017764  112765  000113  000001                  MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
017772  004737  040252                          JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
                                        ;DO SEEK TO CYLINDER LC
017776  013765  005734  000002                  MOV     LC,P.CYLN(R5)   ;SET CYLINDER = LC
020004  004737  032330                          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.
                                        ;*
                                        ;;*****************************************************************
020010  000004                          TST2:   SCOPE
020012  012737  000002  001324                  MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
020020  004737  030416                          JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
020024  004737  030464                          JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
020030  000137  020070                          JMP     TST3            ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
020034  004737  027450                          JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
020040  004737  025766                          JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
                                        ;DO SEEK TO CYLINDER FC
020044  013765  005732  000002                  MOV     FC,P.CYLN(R5)   ;SET CYL = FC
020052  004737  032330                          JSR     PC,SEEKER       ;SEEK TO FC
                                        ;DO SEEK TO CYLINDER LC
020056  013765  005734  000002                  MOV     LC,P.CYLN(R5)   ;SET CYL = LC
020064  004737  032330                          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.
                                        ;*
```

```
5395
5396  020070  000004                        ;;*****************************************************************
5397  020072  012737  000003  001324  TST3: SCOPE
5398  020100  004737  030416                MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5399  020104  004737  030464                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5400  020110  000137  020176                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5401                                        JMP     TST4            ;JUMP TO NEXT TEST
5402  020114  004737  027450          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5403  020120  004737  025766                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5404                                        JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5405  020124  005065  000002          ;SEEK TO CYLINDER 0
5406  020130  004737  032330                CLR     P.CYLN(R5)      ;SET CYL = 0
5407                                        JSR     PC,SEEKER       ;SEEK TO CYL 0
5408  020134  005265  000002          ;SEEK TO CYLINDERS 1,2,4,10,20,40,100,200,400 (FORWARD DIRECTION)
5409  02 140  004737  032330                INC     P.CYLN(R5)      ;INITIALIZE CYL TO 1
5410  020144  006365  000002          2$:   JSR     PC,SEEKER       ;SEEK TO CURRENT CYL
5411  020150  033765  017714  000002        ASL     P.CYLN(R5)      ;GET NEXT CYL NO.
5412  020156  001770                        BIT     HOLD2,P.CYLN(R5) ;SEE IF DONE WITH FORWARD SEEKS
5413                                        BEQ     2$              ;BR IF NOT DONE YET
5414  020160  006265  000002          ;SEEK TO CYLINDERS 400,200,100,40,20,10,4,2,1,0 (REVERSE DIRECTION)
5415  020164  004737  032330          4$:   ASR     P.CYLN(R5)      ;GET NEXT CYL NO.
5416  020170  005765  000002                JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
5417  020174  001371                        TST     P.CYLN(R5)      ;SEE IF DONE IN REVERSE DIRECTION
5418                                        BNE     4$              ;BR IF NOT DONE YET
5419
5420
5421                                  ;;*****************************************************************
5422                                  ;*TEST 4        CYLINDER ADDRESS CROSSTALK TEST
5423                                  ;*
5424                                  ;*      THIS TEST PERFORMS SEEKS TO CYLINDERS WHOSE ADDRESSES
5425                                  ;*      ARE SUSCEPTIBLE TO BIT CROSSTALK WITHIN THE CYLINDER
5426                                  ;*      ADDRESSING LOGIC. THE CYLINDER ADDRESS BITS SEQUENCE
5427                                  ;*      AS FOLLOWS :
5428                                  ;*              000 000 000
5429                                  ;*              011 111 111
5430                                  ;*              000 000 000
5431                                  ;*              011 111 110
5432                                  ;*              000 000 000
5433                                  ;*              011 111 101
5434                                  ;*              000 000 000
5435                                  ;*                  .
5436                                  ;*                  .
5437                                  ;*                  .
5438                                  ;*              000 000 000
5439                                  ;*              010 111 111
5440                                  ;*              000 000 000
5441                                  ;*              101 111 111
5442                                  ;*
5443                                  ;*****************************************************************
5444  020176  000004                  TST4:   SCOPE
5445  020200  012737  000004  001324          MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5446  020206  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5447  020212  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5448  020216  000137  020342                  JMP     TST5            ;JUMP TO NEXT TEST
5449                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5450  020222  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
```

```
5451  020226  004737  025766              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5452                                ;SEEK TO CYL 0
5453  020232  005065  000002              CLR     P.CYLN(R5)      ;SET CYL = 0
5454  020236  004737  032330              JSR     PC,SEEKER       ;SEEK TO CYLINDER 0
5455                                ;SEEK TO CYL 377 (OCT)
5456  020242  012765  000377  000002      MOV     #377,P.CYLN(R5) ;SET CYL = 377
5457  020250  004737  032330              JSR     PC,SEEKER       ;SEEK TO CYL 377
5458                                ;SEEK TO REMAINING CROSSTALK CYLINDERS
5459  020254  012737  000001  005514      MOV     #1,CYLNDR       ;INITIALIZE BIT MASK
5460  020262  005065  000002          2$: CLR     P.CYLN(R5)      ;SET CYL = 0
5461  020266  004737  032330              JSR     PC,SEEKER       ;SEEK TO CYL 0
5462  020272  032737  000200  005514      BIT     #BIT07,CYLNDR   ;SEE IF ABOUT TO DO LAST SEEK
5463  020300  001013                      BNE     4$              ;BR IF THIS WILL BE LAST SEEK
5464  020302  012765  000377  000002      MOV     #377,P.CYLN(R5) ;SET CYL = 377
5465  020310  043765  005514  000002      BIC     CYLNDR,P.CYLN(R5) ;ZERO A BIT TO PROMOTE CROSSTALK
5466  020316  004737  032330              JSR     PC,SEEKER       ;SEEK TO THIS CYLINDER
5467  020322  006337  005514              ASL     CYLNDR          ;SHIFT BIT MASK
5468  020326  000755                      BR      2$              ;GO SEEK TO CYL 0 AGAIN
5469  020330  012765  000577  000002  4$: MOV     #577,P.CYLN(R5) ;SET CYL = 577
5470  020336  004737  032330              JSR     PC,SEEKER       ;SEEK TO CYL 577
5471
5472
5473
5474                                ;;*****************************************************************
5475                                ;*TEST 5        INCREMENT/DECREMENT SEEK TEST
5476                                ;*
5477                                ;*      IN THIS TEST SEEKS ARE DONE IN INCREMENTS OF IC CYLINDERS
5478                                ;*      STARTING AT CYL FC, AND ENDING AT OR BEYOND LC. THEN, THE
5479                                ;*      SEEKS ARE DONE IN REVERSE, BACK TO FC. IF FC IS CHOSEN > LC,
5480                                ;*      THE SEEKS WILL PROCEED IN THE OPPOSITE DIRECTION.
5481                                ;*
5482                                ;;*****************************************************************
5483  020342  000004                TST5:   SCOPE
5484  020344  012737  000005  001324      MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5485  020352  004737  030416              JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5486  020356  004737  030464              JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5487  020362  000137  020546              JMP     TST6            ;JUMP TO NEXT TEST
5488                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5489  020366  004737  027450              JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5490  020372  004737  025766              JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5491                                ;SEEK IN INCREMENTS, FROM FC TO LC
5492  020376  013737  005732  005522      MOV     FC,NCYL1        ;INITIALIZE NCYL1 TO FC
5493  020404  013765  005522  000002  2$: MOV     NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5494  020412  004737  032330              JSR     PC,SEEKER       ;SEEK TO NCYL1
5495  020416  023737  005732  005734      CMP     FC,LC           ;SEE IF FC > LC
5496  020424  003010                      BGT     4$              ;BR IF FC > LC
5497  020426  063737  005736  005522      ADD     IC,NCYL1        ;INCREMENT NCYL1
5498  020434  023737  005734  005522      CMP     LC,NCYL1        ;SEE IF LC EXCEEDED
5499  020442  002360                      BGE     2$              ;BR IF NOT YET
5500  020444  000407                      BR      6$              ;LC REACHED - DO REVERSE SEEKS
5501  020446  163737  005736  005522  4$: SUB     IC,NCYL1        ;DECREMENT NCYL1
5502  020454  023737  005734  005522      CMP     LC,NCYL1        ;SEE IF LC EXCEEDED
5503  020462  003750                      BLE     2$              ;BR IF NOT YET
5504                                ;SEEK IN INCREMENTS, BACK TO FC
5505  020464  023737  005732  005734  6$: CMP     FC,LC           ;SEE IF FC >LC
5506  020472  003010                      BGT     8$              ;BR IF FC > LC
```

```
5507  020474  163737  005736  005522          SUB     IC,NCYL1          ;DECREMENT NCYL1
5508  020502  023737  005732  005522          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5509  020510  003410                          BLE     10$               ;BR IF NOT YET
5510  020512  000415                          BR      12$               ;FC REACHED - ALL DONE
5511  020514  063737  005736  005522  8$:     ADD     IC,NCYL1          ;INCREMENT NCYL1
5512  020522  023737  005732  005522          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5513  020530  002406                          BLT     12$               ;FC REACHED - ALL DONE
5514  020532  013765  005522  000002  10$:    MOV     NCYL1,P.CYLN(R5)  ;SET CYL = NCYL1
5515  020540  004737  032330                  JSR     PC,SEEKER         ;SEEK TO NCYL1
5516  020544  000747                          BR      6$                ;CONTINUE
5517  020546                          12$:
5518
5519
5520                                  ;;*****************************************************************
5521                                  ;*TEST 6        OSCILLATING SEEK TEST
5522                                  ;*
5523                                  ;*      IN THIS TEST, SEEKS OF INCREASING LENGTH ARE DONE FROM FC
5524                                  ;*      TO NCYL1, AND BACK TO FC. NCYL1 IS INCREMENTED BY IC AFTER
5525                                  ;*      EACH SEEK, AND THE SEEKS TO NCYL1 ALWAYS PROCEED FROM FC
5526                                  ;*      TOWARD LC (FC > LC IS PERMISSIBLE). WHEN LC IS REACHED OR
5527                                  ;*      EXCEEDED, THE SAME SEEKS ARE DONE IN REVERSE ORDER, UNTIL
5528                                  ;*      NCYL1 EXCEEDS FC.
5529                                  ;*
5530                                  ;;*****************************************************************
5531  020546  000004                  TST6:   SCOPE
5532  020550  012737  000006  001324          MOV     #6,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
5533  020556  004737  030416                  JSR     PC,SETTUP         ;SET UP FOR LOOP ON ERROR
5534  020562  004737  030464                  JSR     PC,CHKITR         ;SEE IF ITER. NO. = 0 FOR THIS TEST
5535  020566  000137  020776                  JMP     TST7              ;JUMP TO NEXT TEST
5536                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5537  020572  004737  027450                  JSR     PC,INITSS         ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5538  020576  004737  025766                  JSR     PC,PREPKB         ;PREPARE FOR POSSIBLE KBD INPUT
5539                          ;DO OSCILLATING SEEKS TOWARD LC
5540  020602  013737  005732  005522          MOV     FC,NCYL1          ;INITIALIZE NCYL1 TO FC
5541  020610  013765  005522  000002  2$:     MOV     NCYL1,P.CYLN(R5)  ;SET CYL = NCYL1
5542  020616  004737  032330                  JSR     PC,SEEKER         ;SEEK TO NCYL1
5543  020622  013765  005732  000002          MOV     FC,P.CYLN(R5)     ;SET CYL = FC
5544  020630  004737  032330                  JSR     PC,SEEKER         ;SEEK TO FC
5545  020634  023737  005732  005734          CMP     FC,LC             ;SEE IF FC > LC
5546  020642  003010                          BGT     4$                ;BR IF FC > LC
5547  020644  063737  005736  005522          ADD     IC,NCYL1          ;INCREMENT NCYL1
5548  020652  023737  005734  005522          CMP     LC,NCYL1          ;SEE IF LC EXCEEDED
5549  020660  002353                          BGE     2$                ;BR IF NOT YET
5550  020662  000407                          BR      6$                ;LC REACHED - DO REVERSE SEEKS
5551  020664  163737  005736  005522  4$:     SUB     IC,NCYL1          ;DECREMENT NCYL1
5552  020672  023737  005734  005522          CMP     LC,NCYL1          ;SEE IF LC EXCEEDED
5553  020700  003743                          BLE     2$                ;BR IF NOT YET
5554                          ;DO OSCILLATING SEEKS TOWARD FC
5555  020702  023737  005732  005734  6$:     CMP     FC,LC             ;SEE IF FC >LC
5556  020710  003010                          BGT     8$                ;BR IF FC > LC
5557  020712  163737  005736  005522          SUB     IC,NCYL1          ;DECREMENT NCYL1
5558  020720  023737  005732  005522          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
5559  020726  003410                          BLE     10$               ;BR IF NOT YET
5560  020730  000422                          BR      12$               ;FC REACHED - ALL DONE
5561  020732  063737  005736  005522  8$:     ADD     IC,NCYL1          ;INCREMENT NCYL1
5562  020740  023737  005732  005522          CMP     FC,NCYL1          ;SEE IF FC EXCEEDED
```

```
5563  020746  002413                          BLT     12$             ;FC REACHED - ALL DONE
5564  020750  013765  005522  000002  10$:    MOV     NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5565  020756  004737  032330                  JSR     PC,SEEKER       ;SEEK TO NCYL1
5566  020762  013765  005732  000002          MOV     FC,P.CYLN(R5)   ;SET CYL = FC
5567  020770  004737  032330                  JSR     PC,SEEKER       ;SEEK TO FC
5568  020774  000742                          BR      6$              ;CONTINUE
5569  020776                          12$:
5570
5571
5572                                  ;;*****************************************************************
5573                                  ;*TEST 7        CONVERGING/DIVERGING SEEK TEST
5574                                  ;*
5575                                  ;*      THIS TEST PERFORMS SEEKS WHICH EXERCISE CYLINDER DIFFERENCE
5576                                  ;*      VALUES. INITIALLY, NCYL1 = FC AND NCYL2 = LC. A SEEK IS
5577                                  ;*      DONE TO NCYL1 AND THEN TO NCYL2. THEN, NCYL1 IS INCREMENTED
5578                                  ;*      BY IC IN THE DIRECTION OF LC, AND NCYL2 IS INCREMENTED BY
5579                                  ;*      IC IN THE DIRECTION OF FC, AND SEEKS ARE DONE TO NCYL1
5580                                  ;*      AND NCYL2 AGAIN. THIS SEEKING IS CONTINUED WITH INCREMENTED
5581                                  ;*      VALUES OF NCYL1 AND NCYL2, UNTIL NCYL1 EXCEEDS LC AND NCYL2
5582                                  ;*      EXCEEDS FC. (NOTE : FC > LC IS PERMISSIBLE.)
5583                                  ;*      THIS TEST CAUSES SEEKING TO CONVERGING AND THEN DIVERGING
5584                                  ;*      CYLINDER VALUES.
5585                                  ;*
5586                                  ;*****************************************************************
5587  020776  000004                  TST7:   SCOPE
5588  021000  012737  000007  001324          MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
5589  021006  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5590  021012  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5591  021016  000137  021154                  JMP     TST10           ;JUMP TO NEXT TEST
5592                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5593  021022  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5594  021026  004737  025766                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5595  021032  013737  005732  005522          MOV     FC,NCYL1        ;SET NCYL1 = FC
5596  021040  013737  005734  005524          MOV     LC,NCYL2        ;SET NCYL2 = LC
5597  021046  013765  005522  000002  2$:     MOV     NCYL1,P.CYLN(R5) ;SET CYL = NCYL1
5598  021054  004737  032330                  JSR     PC,SEEKER       ;SEEK TO NCYL1
5599  021060  013765  005524  000002          MOV     NCYL2,P.CYLN(R5) ;SET CYL = NCYL2
5600  021066  004737  032330                  JSR     PC,SEEKER       ;SEEK TO NCYL2
5601  021072  023737  005732  005734          CMP     FC,LC           ;SEE IF FC > LC
5602  021100  003013                          BGT     4$              ;BR IF FC > LC
5603  021102  063737  005736  005522          ADD     IC,NCYL1        ;INCREMENT NCYL1
5604  021110  163737  005736  005524          SUB     IC,NCYL2        ;DECREMENT NCYL2
5605  021116  023737  005734  005522          CMP     LC,NCYL1        ;SEE IF LIMITS EXCEEDED
5606  021124  002350                          BGE     2$              ;BR IF NOT YET
5607  021126  000412                          BR      6$              ;LIMITS REACHED - ALL DONE
5608  021130  163737  005736  005522  4$:     SUB     IC,NCYL1        ;DECREMENT NCYL1
5609  021136  063737  005736  005524          ADD     IC,NCYL2        ;INCREMENT NCYL2
5610  021144  023737  005734  005522          CMP     LC,NCYL1        ;SEE IF LIMITS EXCEEDED
5611  021152  003735                          BLE     2$              ;BR IF NOT YET
5612  021154                          6$:
5613
5614
5615
5616                                  ;;*****************************************************************
5617                                  ;*TEST 10       PSEUDO-RANDOM SEEK TEST
5618                                  ;*
```

```
                                        ;*      THIS TEST PERFORMS A SEEK TO A PSEUDO-RANDOMLY CHOSEN
                                        ;*      CYLINDER, WHICH IS WITHIN THE RANGE (0-632).
                                        ;*
                                        ;;*****************************************************************
021154  000004                        TST10:  SCOPE
021156  012737  000010  001324              MOV     #10,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021164  004737  030416                       JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021170  004737  030464                       JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
021174  000137  021226                       JMP     TST11           ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
021200  004737  027450                       JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
021204  004737  025766                       JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
021210  004737  032510                       JSR     PC,RNDADR       ;SELECT A PSEUDO-RANDOM CYLINDER
021214  013765  005514  000002               MOV     CYLNDR,P.CYLN(R5) ;SET CYL VALUE
021222  004737  032330                       JSR     PC,SEEKER       ;SEEK TO THIS CYLINDER
                                        ;;*****************************************************************
                                        ;*TEST 11       MAXIMUM VELOCITY REVERSAL SEEK TEST
                                        ;*
                                        ;*      THIS TEST PERFORMS A SEEK TO CYLINDER 0, FOLLOWED BY A
                                        ;*      SEEK TO CYL 201 (OCT), AND THEN A RETURN SEEK TO 0. THIS
                                        ;*      OPERATION REQUIRES THE HEADS TO DECELERATE UPON REACHING
                                        ;*      THE POINT OF MAXIMUM VELOCITY, AND INDUCES HEATING IN
                                        ;*      THE SERVO MECHANISM.
                                        ;*
                                        ;;*****************************************************************
021226  000004                        TST11:  SCOPE
021230  012737  000011  001324              MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021236  004737  030416                       JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021242  004737  030464                       JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
021246  000137  021314                       JMP     TST12           ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
021252  004737  027450                       JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
021256  004737  025766                       JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
021262  005065  000002                       CLR     P.CYLN(R5)      ;SET CYL = 0
021266  004737  032330                       JSR     PC,SEEKER       ;SEEK TO CYL 0
021272  012765  000201  000002               MOV     #201,P.CYLN(R5) ;SET CYL = 201 (OCT)
021300  004737  032330                       JSR     PC,SEEKER       ;SEEK TO CYL 201
021304  005065  000002                       CLR     P.CYLN(R5)      ,SET CYL = 0
021310  004737  032330                       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
```

```
                                        ;*      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.
                                        ;*
                                        ;;*****************************************************************
021314  000004                    TST12:  SCOPE
021316  012737  000012  001324            MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
021324  004737  03C416                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
021330  004737  030464                    JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
021334  000137  021554                    JMP     TST13           ;JUMP TO NEXT TEST
                                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
021340  004737  027450                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
021344  004737  025766                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
021350  012737  000001  005514            MOV     #1,CYLNDR       ;INIT. CURRENT CYL TO 1
021356  013737  005774  001276            MOV     SM,$TMP6        ;INIT. STALLS TO SM
                                        ;DO INCREASING SEEKS
021364  012737  000012  005542    2$:     MOV     #10.,SCRACH     ;INIT. LOOP COUNTER TO 10(DEC)
021372  013737  001276  005512            MOV     $TMP6,STALLS    ;SET CURRENT STALL IN STALL WORD
021400  005065  000002            4$:     CLR     P.CYLN(R5)      ;SET CYL = 0
021404  004737  032330                    JSR     PC,SEEKER       ;SEEK TO CYL 0
021410  013765  005514  000002            MOV     CYLNDR,P.CYLN(R5) ;SET CURRENT CYLINDER
021416  004737  032330                    JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
021422  005337  005542                    DEC     SCRACH          ;DECREMENT LOOP COUNTER
021426  001364                            BNE     4$              ;BR IF NOT 5 TIMES YET
021430  000241                            CLC                     ;CLEAR GARBAGE BIT
021432  006037  001276                    ROR     $TMP6           ;DIVIDE STALLS BY 2
021436  006037  001300                    ROR     $TMP7           ;SAVE THE DROPPED STALL BIT
021442  006337  005514                    ASL     CYLNDR          ;DOUBLE THE CURRENT CYLINDER NUMBER
021446  032737  001000  005514            BIT     #BIT09,CYLNDR   ;SEE IF DONE WITH INCREASING SEEKS YET
021454  001743                            BEQ     2$              ;BR IF NOT DONE YET
                                        ;DO DECREASING SEEKS
021456  006237  005514            6$:     ASR     CYLNDR          ;DIVIDE CURRENT CYLINDER BY 2
021462  006137  001300                    ROL     $TMP7           ;GET A SAVED STALL BIT INTO CARRY
021466  006137  001276                    ROL     $TMP6           ;DOUBLE THE STALL
021472  012737  000012  005542            MOV     #10.,SCRACH     ;INIT. LOOP COUNTER TO 10(DEC)
021500  013737  001276  005512            MOV     $TMP6,STALLS    ;SET CURRENT STALL IN STALL WORD
021506  005065  000002            8$:     CLR     P.CYLN(R5)      ;SET CYL = 0
021512  004737  032330                    JSR     PC,SEEKER       ;SEEK TO CYL 0
021516  013765  005514  000002            MOV     CYLNDR,P.CYLN(R5) ;SET CURRENT CYL NUMBER
021524  004737  032330                    JSR     PC,SEEKER       ;SEEK TO CURRENT CYLINDER
021530  005337  005542                    DEC     SCRACH          ;DECREMENT LOOP COUNTER
021534  001364                            BNE     8$              ;BR IF NOT 5 TIMES YET
021536  032737  000002  005514            BIT     #BIT01,CYLNDR   ;SEE IF DONE YET
021544  001744                            BEQ     6$              ;BR IF NOT DONE YET
021546  013737  005772  005512            MOV     ST,STALLS       ;RESTORE DESIRED NO. OF UNIT STALLS


                                        ;;*****************************************************************
                                        ;*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.
                                        ;;*****************************************************************
021554  000004                    TST13:  SCOPE
```

```
 5731  021556  012737  000013  001324          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 5732  021564  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
 5733  021570  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
 5734  021574  000137  022366                  JMP     TST14           ;JUMP TO NEXT TEST
 5735                                  ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
 5736  021600  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
 5737  021604  004737  025766                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
 5738  021610  105737  003133                  TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
 5739  021614  001002                          BNE     26$             ;BR IF TEST ALLOWED
 5740  021616  000177  000004                  JMP     @1$             ;SKIP TO NEXT TEST
 5741  021622  004737  032646          26$:    JSR     PC,CALBRT       ;CALIBRATE SOFTWARE TIMER
 5742  021626                          1$:
 5743  021626  022366                          TST14                   ;ERROR RETURN ADDR FOR CALBRT
 5744  02.630  004737  033140                  JSR     PC,INIVRB       ;INIT VARIABLES TO BE USED
 5745  021634  105037  003120                  CLRB    TSTING          ;INHIBIT ^C, ^Z ESCAPE
 5746  021640  005003                          CLR     R3              ;INIT MEASUREMENT COUNTER
 5747                                  ;READ HEADER TO INITIALIZE FORMAT BIT IN CONTROLLER
 5748  021642  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5) ;SET FORMAT BIT = 0
 5749  021650  112765  000125  000001          MOVB    #RDHEAD,P.CMND(R5) ;SET READ HDR COMMAND
 5750  021656  004737  040252                  JSR     PC,DRVCAL       ;DO READ HDR TO SET FORMAT TO 0
 5751  021662  112737  000001  003120          MOVB    #1,TSTING       ;ALLOW ^C, ^Z ESCAPE
 5752                                  ;CHANGE FORMAT AND READ HDR TO FIND INDEX
 5753  021670  005000                  2$:     CLR     R0              ;INIT RK06 INTR INDICATOR
 5754  021672  012777  022350  161140          MOV     #RHEDHD,@RKVEC  ;SET SPECIAL RK06 HANDLER ADDRESS
 5755  021700  005001                          CLR     R1              ;INIT TIME-OUT INDICATOR
 5756  021702  152765  000020  000007          BISB    #B.CFMT,P.CS1H(R5) ;COMPLEMENT FORMAT BIT TO 1
 5757  021710  052765  010000  000016          BIS     #CFMT,P.CS1(R5) ;COMPLEMENT FORMAT BIT TO 1
 5758  021716  004737  040542                  JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
 5759  021722  013712  017722                  MOV     HOLD5,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
 5760  021726  005700                  6$:     TST     R0              ;SEE IF RK06 INTR REC'D YET
 5761  021730  001012                          BNE     8$              ;BR IF RK06 INTR REC'D
 5762  021732  005201                          INC     R1              ;INCREMENT TIME-OUT INDICATOR
 5763  021734  001374                          BNE     6$              ;BR IF NO TIME-OUT
 5764  021736  004737  041650          7$:     JSR     PC,REPSUP       ;GET COMMAND FOR REPORT
 5765  021742  004737  044316                  JSR     PC,TOPROC       ;GATHER STATUS
 5766  021746  104105                          ERROR   105             ;TIMED-OUT ON READ HEADER
 5767  021750  000572                          BR      36$             ;GO TO EXIT
 5768  021752  104410          28$:    RESREG                          ;RESTORE R0-R5
 5769  021754  000770                          BR      7$              ;TAKE ERROR EXIT
 5770                                  ;CHANGE FORMAT AND READ HDR TO MEASURE TIME TO NEXT INDEX
 5771  021756  005000                  8$:     CLR     R0              ;INIT RK06 INTERRUPT INDICATOR
 5772  021760  005004                          CLR     R4              ;INIT HI CYCLE COUNT
 5773  021762  010501                          MOV     R5,R1           ;SAVE PARAM BLK ADDRESS
 5774  021764  010537  005542                  MOV     R5,SCRACH
 5775  021770  005005                          CLR     R5
 5776  021772  142761  000020  000007          BICB    #B.CFMT,P.CS1H(R1) ;COMPLEMENT FORMAT BIT TO 0
 5777  022000  042761  010000  000016          BIC     #CFMT,P.CS1(R1) ;COMPLEMENT FORMAT BIT TO 0
 5778  022006  016162  000016  000000          MOV     P.CS1(R1),RKCS1(R2) ;ISSUE READ HDR TO FIND NEXT INDEX
 5779  022014  004737  033104                  JSR     PC,TIMER        ;MEASURE THE TIME TO NEXT INDEX
 5780  022020  021736                          7$                      ;ERROR RETURN ADDRESS
 5781                                  ;CONVERT MEAS'D CYCLES TO TIME AND CHECK AGAINST LIMITS
 5782  022022  104407                          SAVREG                  ;SAVE R0-R5
 5783  022024  004737  034072                  JSR     PC,CNVTIM       ;CONVERT MEAS'D CYCLES TO TIME
 5784  022030  021752                          28$                     ;ERROR RETURN ADDRESS
 5785  022032  060337  003176                  ADD     R3,SUMLO1       ;ADD CONVERTED TIME TO SUM
 5786  022036  005537  003200                  ADC     SUMHI1
```

```
5787  022042  060237  003200                ADD     R2,SUMHI1
5788  022046  004737  036266                JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
5789                                ;COMPARE R2-R3 AGAINST CURRENT MIN AND MAX MEASUREMENTS
5790  022052  020237  003214                CMP     R2,MINIH1       ;COMP HI BITS TO HI MIN
5791  022056  101010                        BHI     32$             ;BR IF > MIN
5792  022060  103403                        BLO     30$             ;BR IF < MIN
5793  022062  020337  003212                CMP     R3,MINIL1       ;COMP LO BITS TO LO MIN
5794  022066  103004                        BHIS    32$             ;BR IF > OR = MIN
5795  022070  010237  003214        30$:    MOV     R2,MINIH1       ;SET NEW MIN
5796  022074  010337  003212                MOV     R3,MINIL1
5797  022100  020237  003220        32$:    CMP     R2,MAXIH1       ;COMP HI BITS TO HI MAX
5798  022104  103410                        BLO     14$             ;BR IF < MAX
5799  022106  101003                        BHI     34$             ;BR IF > MAX
5800  022110  020337  003216                CMP     R3,MAXIL1       ;COMP LO BITS TO LO MAX
5801  022114  101404                        BLOS    14$             ;BR IF < OR = MAX
5802  022116  010237  003220        34$:    MOV     R2,MAXIH1       ;SET NEW MAX
5803  022122  010337  003216                MOV     R3,MAXIL1
5804                                ;COMPARE MEAS'D TIME TO SPECIFIED LIMITS
5805  022126  020227  000000        14$:    CMP     R2,#MINHI1      ;COMPARE HI BITS TO HI MINIMUM
5806  022132  101006                        BHI     18$             ;BR IF > MIN
5807  022134  103403                        BLO     16$             ;BR IF < MIN
5808  022136  020327  057467                CMP     R3,#MINLO1      ;COMPARE LO BITS TO LO MIN
5809  022142  103002                        BHIS    18$             ;BR IF > OR = MIN
5810  022144  005237  003170        16$:    INC     BLWMIN          ;INCREMENT COUNT OF TIMES BELOW MIN
5811  022150  020227  000000        18$:    CMP     R2,#MAXHI1      ;COMPARE HI BITS TO HI MAX
5812  022154  103406                        BLO     22$             ;BR IF < MAX
5813  022156  101003                        BHI     20$             ;BR IF > MAX
5814  022160  020327  062031                CMP     R3,#MAXLO1      ;COMPARE LO BITS TO LO MAX
5815  022164  101402                        BLOS    22$             ;BR IF < OR = MAX
5816  022166  005237  003172        20$:    INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE MAX
5817  022172  104410                22$:    RESREG                  ;RESTORE R0-R5
5818  022174  005203                        INC     R3              ;INCREMENT MEASUREMENT COUNTER
5819  022176  013705  005542                MOV     SCRACH,R5       ;RESTORE PARAM BLK ADDRESS
5820  022202  022703  000200                CMP     #128.,R3        ;SEE IF 128 MEASUREMENTS YET
5821  022206  001230                        BNE     2$              ;BR IF NOT DONE WITH MEASUREMENTS YET
5822                                ;DVIDE SUM BY 128. TO GET AVERAGE TIME OF ROTATION
5823  022210  012704  000007                MOV     #7,R4           ;SET LOOP COUNTER = 7
5824  022214  000241                24$:    CLC                     ;DIVIDE DOUBLE-WORD SUM BY 2
5825  022216  006037  003200                ROR     SUMHI1
5826  022222  006037  003176                ROR     SUMLO1
5827  022226  005304                        DEC     R4              ;SEE IF DONE WITH DIVISION
5828  022230  001371                        BNE     24$             ;BR IF NOT DONE YET
5829                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
5830  022232  032737  000010  005770        BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5831  022240  001036                        BNE     36$             ;BR IF REPORTS SHOULD BE INHIBITED
5832  022242  104401  011540                TYPE    ,ROTIMS         ;TYPE "ROTATIONAL TIMES :"
5833  022246  012701  003212                MOV     #MINIL1,R1      ;POINTER TO LO MIN
5834  022252  004737  034144                JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
5835  022256  013746  003170                MOV     BLWMIN,-(SP)    ;GET COUNT OF TIMES BELOW MIN
5836  022262  104405                        TYPDS                   ;CONVERT AND TYPE IT
5837  022264  104401  011750                TYPE    ,BELOW          ;TYPE " OF 128 BELOW MIN OF "
5838  022270  104401  012261                TYPE    ,LIM1           ;TYPE SPEC'D MIN LIMIT
5839  022274  012701  003216                MOV     #MAXIL1,R1      ;POINTER TO LO MAX
5840  022300  004737  034170                JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
5841  022304  013746  003172                MOV     ABVMX1,-(SP)    ;GET COUNT OF TIMES ABOVE MAX
5842  022310  104405                        TYPDS                   ;CONVERT AND TYPE IT
```

```
5843  022312  104401  012005                  TYPE    ,ABOVE          ;TYPE " OF 128 ABOVE MAX OF "
5844  022316  104401  012274                  TYPE    ,LIM2           ;TYPE SPEC'D MAX LIMIT
5845  022322  012701  003176                  MOV     #SUMLO1,R1      ;POINTER TO AVG TIME
5846  022326  004737  034214                  JSR     PC,TYPAVG       ;TYPE AVERAGE TIME
5847  022332  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
5848  022336  012777  045234  160474  36$:    MOV     #I.INTR,@RKVEC  ;RESTORE RK06 HANDLER ADDRESS
5849  022344  000177  177256                  JMP     @1$             ;PROCEED TO NEXT TEST
5850
5851                          ;;*****************************************************************
5852                          ;*SPECIAL INTERRUPT HANDLER FOR READ HEADERS IN ROT. LAT. MEAS. TEST
5853                          ;;*****************************************************************
5854  022350  005200          R4EDHD: INC     R0              ;SET RK06 INTR HANDLER
5855  022352  005762  000000          TST     RKCS1(R2)       ;SEE IF CONTROLLER ERROR SET
5856  022356  100002                  BPL     2$              ;BR IF NO ERROR
5857  022360  000137  045234          JMP     I.INTR          ;LET DRIVER PROCESS THE ERROR
5858  022364  000002          2$:     RTI                     ;RETURN FROM INTR
5859
5860
5861
5862                          ;;*****************************************************************
5863                          ;*TEST 14       ONE CYLINDER SEEK TIME MEASUREMENT
5864                          ;*THIS TEST MEASURES THE TIME REQUIRED TO SEEK BETWEEN 2 AJACENT
5865                          ;*CYLINDERS, BOTH IN THE FORWARD AND REVERSE DIRECTIONS. SEEKS
5866                          ;*ARE DONE TO 0,1,2,...,631,632  AND THEN TO 631,630,...,2,1,0
5867                          ;* AND THE RESULTS ARE TYPED FOR EACH DIRECTION.
5868                          ;*THE SPECIFIED ONE CYL SEEK TIME IS < 8 MILLI-SEC.
5869                          ;* SEEKS ARE TO & FROM CYL 1456 FOR THE RK07
5870                          ;;*****************************************************************
5871  022366  000004          TST14:  SCOPE
5872  022370  012737  000014  001324          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
5873  022376  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5874  022402  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5875  022406  000137  022706                  JMP     TST15           ;JUMP TO NEXT TEST
5876                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5877  022412  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5878  022416  004737  025766                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5879  022422  105737  003133                  TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
5880  022426  001002                  BNE     2$              ;BR IF ALLOWED
5881  022430  000177  000004          1$:     JMP     @4$             ;SKIP TO NEXT TEST
5882  022434  004737  032646          2$:     JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
5883  022440                          4$:
5884  022440  022706                  TST15                   ;ERROR RETURN ADDR FOR CALBRT
5885  022442  004737  033140                  JSR     PC,INIVRB       ;INIT VARIABLES USED
5886  022446  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5887  022454  005065  000002                  CLR     P.CYLN(R5)      ;INIT CYL TO 0
5888  022460  004737  040252                  JSR     PC,DRVCAL       ;DO INITIAL SEEK TO CYL 0
5889                          ;MEASURE FORWARD SEEK TIME
5890  022464  005265  000002          6$:     INC     P.CYLN(R5)      ;INCREMENT CYLINDER BY 1
5891  022470  004737  033246                  JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
5892  022474  022706                  TST15                   ;ERROR RETURN ADDR FOR CALBRT
5893  022476  060137  003176                  ADD     R1,SUMLO1       ;ADD MEAS'D TIME TO FORWARD SUM
5894  022502  005537  003200                  ADC     SUMHI1
5895  022506  060037  003200                  ADD     R0,SUMHI1
5896  022512  004737  036266                  JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
5897                          ;COMPARE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
5898  022516  012703  003212                  MOV     #MINIL1,R3      ;SET POINTER TO CURRENT MIN AND MAX
```

```
5899  022522  004737  033560                JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5900                                ;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
5901  022526  020037  007220                CMP     R0,MAXHI2       ;COMPARE HI BITS TO HI MAX
5902  022532  103406                        BLO     12$             ;BR IF < MAX
5903  022534  101003                        BHI     10$             ;BR IF > MAX
5904  022536  020137  007216                CMP     R1,MAXLO2       ;COMPARE LO BITS TO LO MAX
5905  022542  101402                        BLOS    12$             ;BR IF < OR = MAX
5906  022544  005237  003172          10$:  INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
5907  022550  023765  017706  000002  12$:  CMP     LSTCYL,P.CYLN(R5)       ;SEE IF LAST FORWARD SEEK DONE
5908  022556  001342                        BNE     6$              ;BR IF NOT DONE
5909                                ;MEASURE A REVERSE SEEK TIME
5910  022560  005365  000002          14$:  DEC     P.CYLN(R5)      ;DECREMENT CYL BY 1
5911  022564  004737  033246                JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
5912  022570  022706                        TST15                   ;ERROR RETURN ADDR FOR CALBRT
5913  022572  060137  003202                ADD     R1,SUMLO2       ;ADD MEASURED TIME TO REVERSE SUM
5914  022576  005537  003204                ADC     SUMHI2
5915  022602  060037  003204                ADD     R0,SUMHI2
5916  022606  004737  036266                JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
5917                                ;COMPARE REVERSE TIME TO CURRENT REVERSE MIN AND MAX
5918  022612  012703  003222                MOV     #MINIL2,R3      ;SET POINTER TO CURRENT MIN AND MAX
5919  022616  004737  033560                JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
5920                                ;COMPARE REVERSE TIME TO SPEC'D MAX LIMIT
5921  022622  020037  007220                CMP     R0,MAXHI2       ;COMPARE HI BITS TO HI MAX
5922  022626  103406                        BLO     18$             ;BR IF < MAX
5923  022630  101003                        BHI     16$             ;BR IF > MAX
5924  022632  020137  007216                CMP     R1,MAXLO2       ;COMPARE LO BITS TO LO MAX
5925  022636  101402                        BLOS    18$             ;BR IF < OR = MAX
5926  022640  005237  003174          16$:  INC     ABVMX2          ;INCR COUNT OF TIMES ABOVE SPEC'D MAX
5927  022644  005765  000002          18$:  TST     P.CYLN(R5)      ;SEE IF LAST REVERSE SEEK DONE
5928  022650  001343                        BNE     14$             ;BR IF NOT DONE YET
5929                                ;COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES
5930  022652  013703  017706                MOV     LSTCYL,R3               ;GET NO. OF MEASUREMENTS
5931  022656  004737  033632                JSR     PC,GETAVG       ;COMPUTE AVERAGES
5932                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
5933  022662  032737  000010  005770        BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
5934  022670  001257                        BNE     1$              ;BR IF REPORTS SHOULD BE INHIBITED
5935  022672  104401  011605                TYPE    ,ONECYL         ;TYPE "ONE CYL SEEK TIMES"
5936  022676  012703  012042                MOV     #ABOVE1,R3      ;GET POINTER TO MAX SPEC'D LIMIT MSG
5937  022702  004737  033724                JSR     PC,TYPTMS       ;TYPE TIMING TEST RESULTS
5938
5939
5940
5941                                ;;*****************************************************************
5942                                ;*TEST 15       AVERAGE SEEK TIME MEASUREMENT
5943                                ;*THIS TEST MEASURES THE TRUE AVERAGE SEEK TIME IN BOTH THE FORWARD
5944                                ;*AND REVERSE DIRECTIONS. THE AVERAGE TIME IS CALCULATED FROM THE
5945                                ;*FOLLOWING FORMULA :
5946                                ;*  T AVG = [T1(410)(2)+T2(409)(2)+...+T410(1)(2)]/(410)(410)
5947                                ;*  WHERE TX = THE MEASURED TIME TO SEEK X CYLINDERS.
5948                                ;*FORWARD AND REVERSE TIMES ARE MEASURED AND TYPED,
5949                                ;*SEPARATELY.
5950                                ;*THE SPECIFIED AVERAGE SEEK TIME IS < 38 MILLI-SEC.
5951                                ;* CYL 814 USED FOR THE RK07
5952                                ;;*****************************************************************
5953  022706  000004                TST15:  SCOPE
5954  022710  012737  000015  001324        MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
5955  022716  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
5956  022722  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
5957  022726  000137  023244                  JMP     TST16           ;JUMP TO NEXT TEST
5958                          ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
5959  022732  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
5960  022736  004737  025766                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
5961  022742  105737  003133                  TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
5962  022746  001002                          BNE     2$              ;BR IF ALLOWED
5963  022750  000177  000004          1$:     JMP     @4$             ;SKIP TO NEXT TEST
5964  022754  004737  032646          2$:     JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
5965  022760                          4$:
5966  022760  023244                          TST16                   ;ERROR RETURN ADDR FOR CALBRT
5967  022762  004737  033140                  JSR     PC,INIVRB       ;INIT VARIABLES
5968  022766  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
5969  022774  005065  000002                  CLR     P.CYLN(R5)      ;SET CYL = 0
5970  023000  004737  040252                  JSR     PC,DRVCAL       ;SEEK INITIALLY TO 0
5971  023004  005037  005522                  CLR     NCYL1           ;SET DESTINATION CYLINDER
5972  023010  013737  017710  005542          MOV     LCP1,SCRACH     ;INIT COEFFICIENT TO 411(DEC)/814(DEC) FOR RK07
5973  023016  005237  005522          8$:     INC     NCYL1           ;INCR DESTINATION CYL
5974  023022  005337  005542                  DEC     SCRACH          ;DECREMENT COEFFICIENT
5975  023026  013765  005522  000002          MOV     NCYL1,P.CYLN(R5)
5976  023034  004737  033246                  JSR     PC,TIMSEK       ;MEASURE A FORWARD SEEK TIME
5977  023040  023244                          TST16                   ;ERROR RETURN ADDR FOR CALBRT
5978  023042  004737  033450                  JSR     PC,FDTERM       ;COMPUTE AND ADD A FORWARD TERM
5979  023046  005065  000002                  CLR     P.CYLN(R5)      ;SET CYL TO 0
5980  023052  004737  033246                  JSR     PC,TIMSEK       ;MEASURE A REVERSE SEEK TIME
5981  023056  023244                          TST16                   ;ERROR RETURN ADDR FOR CALBRT
5982  023060  004737  033514                  JSR     PC,RVTERM       ;COMPUTE AND ADD A REVERSE TERM
5983  023064  023737  005522  017706          CMP     NCYL1,LSTCYL    ;SEE IF DONE MEASURING YET
5984  023072  002751                          BLT     8$              ;BR IF NOT DONE YET
5985  023074  104401  011560                  TYPE    ,AVGSEK         ;TYPE "AVERAGE SEEK TIMES"
5986                  ;DIVIDE FORWARD SUM BY (410)(410)/(814)(814) FOR RK07
5987  023100  013700  003176                  MOV     SUMLO1,R0       ;GET DIVIDEND
5988  023104  013701  003200                  MOV     SUMHI1,R1
5989  023110  013702  003202                  MOV     SUMLO2,R2
5990  023114  013703  003204                  MOV     SUMHI2,R3
5991  023120  013704  017724                  MOV     HOLD6,R4                ;GET DIVISOR
5992  023124  013705  017726                  MOV     HOLD7,R5
5993  023130  004737  053210                  JSR     PC,M.DPID       ;PERFORM DIVISION
5994  023134  010237  003200                  MOV     R2,SUMHI1       ;GET FORWARD AVG
5995  023140  010337  003176                  MOV     R3,SUMLO1
5996                  ;DIVIDE REVERSE SUM BY (410)(410)
5997  023144  013700  003216                  MOV     MAXIL1,R0       ;GET DIVIDEND
5998  023150  013701  003220                  MOV     MAXIH1,R1
5999  023154  013702  003226                  MOV     MAXIL2,R2
6000  023160  013703  003230                  MOV     MAXIH2,R3
6001  023164  004737  053210                  JSR     PC,M.DPID       ;PERFORM DIVISION
6002  023170  010237  003204                  MOV     R2,SUMHI2       ;GET REVERSE AVG
6003  023174  010337  003202                  MOV     R3,SUMLO2
6004                  ;TYPE FORWARD AVERAGE
6005  023200  104401  011663                  TYPE    ,FORWRD         ;TYPE "***FORWARD DIRECTION***"
6006  023204  012701  003176                  MOV     #SUMLO1,R1      ;POINTER TO FORWARD AVG
6007  023210  004737  034214                  JSR     PC,TYPAVG       ;TYPE FORWARD AVERAGE
6008  023214  104401  012225                  TYPE    ,SPCDMX         ;TYPE "SPEC'D MAX IS 40000 US"
6009                  ;TYPE REVERSE AVERAGE
6010  023220  104401  011701                  TYPE    ,REVRSE         ;TYPE "***REVERSE DIRECTION***"
```

```
                                MOV     #SUMLO2,R1      ;POINTER TO REVERSE AVG
                                JSR     PC,TYPAVG       ;TYPE REVERSE AVERAGE
                                TYPE    ,SPCDMX         ;TYPE "SPEC'D MAX IS 40000 US"
                                JSR     PC,INITSS       ;INIT THE SUB-SYS

                        ;;**********************************************************************
                        ;*TEST 16       MAXIMUM SEEK TIME MEASUREMENT
                        ;*THIS TEST MEASURES THE TIME  REQUIRED TO SEEK FORM CYL 0 TO
                        ;*CYL 632 OCT. (410 DEC.). THIS TIME REPRESENTS THE MAXIMUM SEEK
                        ;*TIME, AND IT IS MEASURED 128 TIMES IN EACH DIRECTION.  THE
                        ;*THE SPECIFIED MAXIMUM SEEK TIME IS < 75 MILLI-SEC.
                        ;* CYL 814 IS THE LAST CYL FOR THE RK07
                        ;;**********************************************************************
023244  000004         TST16:  SCOPE
023246  012737  000016  001324        MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
023254  004737  030416                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
023260  004737  030464                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
023264  000137  023570                JMP     TST17           ;JUMP TO NEXT TEST
                        ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
023270  004737  027450                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
023274  004737  025766                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
023300  105737  003133                TSTB    DOTIM           ;SEE IF TIMING TESTS ARE ALLOWED
023304  001002                        BNE     2$              ;BR IF ALLOWED
023306  000177  000004        1$:     JMP     @4$             ;SKIP TO NEXT TEST
023312  004737  032646        2$:     JSR     PC,CALBRT       ;CALIBRATE THE SOFTWARE TIMER
023316                        4$:
023316  023570                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
023320  004737  033140                JSR     PC,INIVRB       ;INIT VARIABLES USED
023324  112765  000117  000001        MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
023332  005065  000002                CLR     P.CYLN(R5)      ;INIT CYL TO 0
023336  004737  040252                JSR     PC,DRVCAL       ;DO INITIAL SEEK TO CYL 0
023342  005037  005542                CLR     SCRACH          ;INIT MEASUREMENT COUNT
                        ;MEASURE FORWARD SEEK TIME
023346  013765  017706  000002  6$:   MOV     LSTCYL,P.CYLN(R5)       ;SET CYL = 632 OCT, 1456 FOR RK07
023354  004737  033246                JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
023360  023570                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
023362  060137  003176                ADD     R1,SUMLO1       ;ADD MEAS'D TIME TO FORWARD SUM
023366  005537  003200                ADC     SUMHI1
023372  060037  003200                ADD     R0,SUMHI1
023376  004737  036266                JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
                        ;COMPARE FORWARD TIME TO CURRENT FORWARD MIN AND MAX
023402  012703  003212                MOV     #MINIL1,R3      ;SET POINTER TO CURRENT MIN AND MAX
023406  004737  033560                JSR     PC,CMPTIM       ;COMPARE TO MEAS'D MIN AND MAX
                        ;COMPARE FORWARD TIME TO SPEC'D MAX LIMIT
023412  020037  007224                CMP     R0,MAXHI4       ;COMPARE HI BITS TO HI MAX
023416  103406                        BLO     12$             ;BR IF < MAX
023420  101003                        BHI     10$             ;BR IF > MAX
023422  020137  007222                CMP     R1,MAXLO4       ;COMPARE LO BITS TO LO MAX
023426  101402                        BLOS    12$             ;BR IF < OR = MAX
023430  005237  003172        10$:    INC     ABVMX1          ;INCREMENT COUNT OF TIMES ABOVE SPEC'D MAX
                        ;MEASURE A REVERSE SEEK TIME
023434  005065  000002        12$:    CLR     P.CYLN(R5)      ;SET CYL = 0
023440  004737  033246                JSR     PC,TIMSEK       ;MEASURE A SEEK TIME
023444  023570                        TST17                   ;ERROR RETURN ADDR FOR CALBRT
023446  060137  003202                ADD     R1,SUMLO2       ;ADD MEASURED TIME TO REVERSE S M
```

```
6067  023452  005537  003204                ADC     SUMHI2
6068  023456  060037  003204                ADD     R0,SUMHI2
6069  023462  004737  036266                JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
6070                                ;COMPARE REVERSE TIME TO CURRENT REVERSE MIN AND MAX
6071  023466  012703  003222                MOV     #MINIL2,R3      ;SET POINTER TO CURRENT MIN AND MAX
6072  023472  004737  033560                JSR     PC,CMPTIM       ;COMPARE TO MEAS'T MIN AND MAX
6073                                ;COMPARE REVERSE TIME TO SPEC'D MAX LIMIT
6074  023476  020037  007224                CMP     R0,MAXHI4       ;COMPARE HI BITS TO HI MAX
6075  023502  103406                        BLO     18$             ;BR IF < MAX
6076  023504  101003                        BHI     16$             ;BR IF > MAX
6077  023506  020137  007222                CMP     R1,MAXLO4       ;COMPARE LO BITS TO LO MAX
6078  023512  101402                        BLOS    18$             ;BR IF < OR = MAX
6079  023514  005237  003174          16$:  INC     ABVMX2          ;INCR COUNT OF TIMES ABOVE SPEC'D MAX
6080  023520  005237  005542          18$:  INC     SCRACH          ;INCREMENT MEASUREMENT COUNT
6081  023524  022737  000200  005542        CMP     #128.,SCRACH    ;SEE IF DONE WITH 128 MEASUREMENTS
6082  023532  001305                        BNE     6$              ;BR IF NOT DONE YET
6083                                ;COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES
6084  023534  012703  000200                MOV     #128.,R3        ;GET NO. OF MEASUREMENTS
6085  023540  004737  033632                JSR     PC,GETAVG       ;COMPUTE AVERAGES
6086                                ;TYPE TIMING MEASUREMENT RESULTS (IN DECIMAL)
6087  023544  032737  000010  005770        BIT     #BIT03,CS       ;SEE IF TIMING TEST REPORTS SHOULD BE INHIBITED
6088  023552  001255                        BNE     1$              ;BR IF REPORTS SHOULD BE INHIBITED
6089  023554  104401  011636                TYPE    ,MAXSEK         ;TYPE 'MAXIMUM SEEK TIMES"
6090  023560  012703  012142                MOV     #ABOVE3,R3      ;GET POINTER TO MAX SPEC'D LIMIT MSG
6091  023564  004737  033724                JSR     PC,TYPTMS       ;TYPE TIMING TEST RESULTS
6092
6093
6094
6095                                ;;*****************************************************************
6096                                ;*TEST 17       SECTOR ADDRESSING TEST
6097                                ;*IN THIS TEST, ALL SECTORS OF FC, FT ARE WRITTEN, EACH WITH
6098                                ;*256(DEC) WORDS OF ITS OWN SECTOR NO. + 100(OCT). NEXT, FOR
6099                                ;*EACH SECTOR WRITTEN A WRITE CHECK IS DONE, FOLLOWED BY A
6100                                ;*READ AND SOFTWARE COMPARE OF THE DATA.
6101                                ;;*****************************************************************
6102  023570  000004                TST17:  SCOPE
6103  023572  012737  000017  001324        MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6104  023600  004737  030416                JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6105  023604  004737  030464                JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
6106  023610  000137  024210                JMP     TST20           ;JUMP TO NEXT TEST
6107                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
6108  023614  004737  027450                JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
6109  023620  004737  025766                JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
6110  023624  112765  000123  000001        MOVB    #WRDATA,P.CMND(R5)      ;SET WRITE DATA COMMAND
6111  023632  013765  005732  000002        MOV     FC,P.CYLN(R5)           ;SET CYL = FC
6112  023640  113765  005740  000005        MOVB    FT,P.TRCK(R5)           ;SET TRACK = FT
6113  023646  013737  005732  001174        MOV     FC,$REG5                ;SET PACK ADR FOR ERROR PRINTOUT
6114  023654  013737  005740  001176        MOV     FT,$REG6
6115  023662  005037  001200                CLR     $REG7
6116  023666  105065  000004                CLRB    P.SECT(R5)              ;SET SECTOR = 0
6117  023672  012765  063520  000010        MOV     #RWBUF,P.BALO(R5)       ;SET DATA BUFFER ADDRESS
6118  023700  012765  177400  000012        MOV     #-256.,P.WC(R5)         ;SET WORD CNT FOR 256(DEC) WORDS
6119                                ;SET NUMBER OF SECTORS IN R3
6120  023706  012703  000024                MOV     #20.,R3         ;SET R3 INITIALLY = 20(DEC)
6121  023712  105737  003125                TSTB    FORMAT          ;DETERMINE THE FORMAT
6122  023716  001002                        BNE     4$              ;BR IF 20 SECTOR FORMAT
```

```
6123  023720  012703  000026            MOV     #22.,R3          ;SET R3 = 22(DEC)
6124                                ;WRITE THE CURRENT SECTOR WITH THE SECTOR NUMBER + 100(OCT)
6125  023724  116500  000004        4$: MOVB    P.SECT(R5),R0    ;PUT SECTOR NO. INTO R0
6126  023730  062700  000100            ADD     #100,R0          ;ADD 100(OCT) TO SECTOR NO.
6127  023734  004737  034240            JSR     PC,LODSEC        ;LOAD THE ENTIRE BUFFER WITH SECTOR NO. + 100(OCT)
6128  023740  004737  040252            JSR     PC,DRVCAL        ;WRITE THE SECTOR
6129  023744  105265  000004            INCB    P.SECT(R5)       ;INCREMENT THE SECTOR NO.
6130  023750  120365  000004            CMPB    R3,P.SECT(R5)    ;SEE IF LAST SECTOR WRITTEN YET
6131  023754  001363                    BNE     4$               ;BR IF NOT DONE YET
6132  023756  012737  023756  001110    MOV     #.,$LPERR        ;SET NEW LOOP ON ERROR ADRS
6133  023764  004737  030416            JSR     PC,SETTUP        ;SET UP FOR LOOP ON ERROR
6134  023770  105065  000004            CLRB    P.SECT(R5)       ;INITIALIZE SECTOR TO 0
6135                                ;DO WRITE CHECK OF A SECTOR
6136  023774  116500  000004        8$: MOVB    P.SECT(R5),R0    ;GET SECTOR INTO R0
6137  024000  062700  000100            ADD     #100,R0          ;ADD 100(OCT) TO SECTOR NO.
6138  024004  004737  034240            JSR     PC,LODSEC        ;LOAD BUFFER WITH SECTOR + 100(OCT)
6139  024010  112765  000131  000001    MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6140  024016  004737  040252            JSR     PC,DRVCAL        ;DO A WRITE CHECK
6141                                ;READ THE SECTOR AND COMPARE THE DATA TO SECTOR NUMBER + 100(OCT)
6142  024022  112765  000121  000001    MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
6143  024030  004737  040252            JSR     PC,DRVCAL        ;READ THIS SECTOR INTO MEMORY
6144  024034  012701  063520            MOV     #RWBUF,R1        ;GET ADDR. OF DATA BUF INTO R1
6145  024040  005004                    CLR     R4               ;CLEAR WORD NUMBER
6146  024042  005037  005542            CLR     SCRACH           ;INIT. COMPARE ERROR COUNT
6147  024046  116537  000004  001200    MOVB    P.SECT(R5),$REG7
6148  024054  032737  000002  005504    BIT     #BSERR,RECODE    ;SEE IF BAD SECTOR ERROR OCCURRED
6149  024062  001045                    BNE     22$              ;BR IF YES
6150  024064  020021                10$: CMP     R0,(R1)+         ;COMPARE BUF WORD TO SECTOR + 100(OCT)
6151  024066  001437                    BEQ     20$              ;BR IF NO COMPARE ERROR
6152  024070  005737  005542            TST     SCRACH           ;CHECK ERROR COUNT
6153  024074  001006                    BNE     12$              ;BR IF NOT FIRST ERROR IN SECTOR
6154  024076  105037  062051            CLRB    DH701+38.        ;ADJUST DATA HEADER FOR MSG
6155  024102  012737  000005  063412    MOV     #5,DF25+2        ;ADJ. ERROR DATA WORD COUNT
6156  024110  104034                    ERROR   34               ;TYPE HEADING FOR ERROR MSG
6157  024112  005237  005542        12$: INC     SCRACH           ;INCREMENT ERROR COUNT
6158  024116  032777  000001  155014    BIT     #BIT0,@SWR       ;SEE IF ALL ERRORS SHOULD BE REPORTED
6159  024124  001004                    BNE     14$              ;BR TO REPORT ALL ERRORS
6160  024126  022737  000012  005542    CMP     #10.,SCRACH      ;SEE IF 10(DEC) ERRORS YET
6161  024134  002420                    BLT     22$              ;BR IF > 10.
6162  024136  010437  001202        14$: MOV     R4,$REG10        ;WORD NUMBER
6163  024142  010037  001204            MOV     R0,$REG11        ;GOOD DATA
6164  024146  016137  177776  001206    MOV     -2(R1),$REG12    ;BAD DATA
6165  024154  104063                    ERROR   63               ;TYPE GOOD AND BAD DATA
6166  024156  032777  000100  154754    BIT     #BIT6,@SWR       ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
6167  024164  001004                    BNE     22$              ;BR IF JUST 1 ERROR SHOULD BE REPORTED
6168  024166  005204                20$: INC     R4               ;INCREMENT WORD NUMBER
6169  024170  022704  000400            CMP     #256.,R4         ;SEE IF DONE YET
6170  024174  001333                    BNE     10$              ;BR IF NOT DONE COMPARING YET
6171  024176  105265  000004        22$: INCB    P.SECT(R5)       ;INCREMENT SECTOR NO.
6172  024202  120365  000004            CMPB    R3,P.SECT(R5)    ;SEE IF ALL SECTORS CHECKED YET
6173  024206  001272                    BNE     8$               ;BR IF NOT DONE YET
6174
6175
6176
6177                                ;;*******************************************************************
6178                                ;*TEST 20        TRACK ADDRESSING TEST
```

```
6179                                    ;*IN THIS TEST, SECTOR FS OF CYL FC IS WRITTEN WITH 256 (DEC) WORDS
6180                                    ;*OF THE TRACK NO. + 100(OCT), FOR EACH OF TRACKS 0,1,2. THEN, A
6181                                    ;*WRITE CHECK OF EACH SECTOR IS DONE TO VERIFY THE WRITES. THEN, EACH
6182                                    ;*OF THE 3 SECTORS IS RE-WRITTEN, AND AFTER EACH WRITE ALL OF THE
6183                                    ;*THREE SECTORS ARE WRITE-CHECKED TO DETECT TRACK ADDRESSING PROBLEMS.
6184                                    ;;*****************************************************************
6185  024210  000004                    TST20:  SCOPE
6186  024212  012737  000020  001324            MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
6187  024220  004737  030416                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6188  024224  004737  030464                    JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
6189  024230  000137  024444                    JMP     TST21           ;JUMP TO NEXT TEST
6190                                    ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
6191  024234  004737  027450                    JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
6192  024240  004737  025766                    JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
6193  024244  112765  000123  000001            MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
6194  024252  013765  005732  000002            MOV     FC,P.CYLN(R5)   ;SET CYL = FC
6195  024260  105065  000005                    CLRB    P.TRCK(R5)      ;INITIALIZE TRACK NO. TO 0
6196  024264  113765  005516  000004            MOVB    FS,P.SECT(R5)   ;SET SECTOR =FS
6197  024272  012765  063520  000010            MOV     #RWBUF,P.BALO(R5) ;SET DATA BUFFER ADDRESS
6198  024300  012765  177400  000012            MOV     #-256.,P.WC(R5) ;SET WORD COUNT FOR 256(DEC) WORDS
6199                                    ;WRITE THE TRACK NO. + 100(OCT) AT SECTOR FS OF TRACKS 0,1,2
6200  024306  116500  000005            4$:     MOVB    P.TRCK(R5),R0   ;SET TRACK NO. IN R0
6201  024312  062700  000100                    ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
6202  024316  004737  034240                    JSR     PC,LODSEC       ;LOAD BUFFER WITH TRACK NO. + 100(OCT)
6203  024322  004737  040252                    JSR     PC,DRVCAL       ;WRITE THE SECTOR
6204  024326  105265  000005                    INCB    P.TRCK(R5)      ;INCREMENT THE TRACK NO.
6205  024332  122765  000003  000005            CMPB    #3,P.TRCK(R5)   ;SEE IF ALL 3 TRACKS WRITTEN YET
6206  024340  001362                            BNE     4$              ;BR IF NOT DONE YET
6207                                    ;DO A WRITE-CHECK OF THE 3 SECTORS WRITTEN, TO VERIFY THE DATA
6208  024342  012737  024342  001110            MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
6209  024350  004737  030416                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6210  024354  004737  034262                    JSR     PC,TRKCHK
6211                                    ;WRITE A SECTOR AND WRITE-CHECK ALL 3 SECTORS
6212  024360  012737  024360  001110            MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
6213  024366  004737  030416                    JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6214  024372  105065  000005                    CLRB    P.TRCK(R5)      ;INIT TRACK TO 0
6215  024376  116500  000005            6$:     MOVB    P.TRCK(R5),R0   ;GET TRACK NO. IN R0
6216  024402  062700  000100                    ADD     #100,R0         ;ADD 100(OCT) TO TRACK NO.
6217  024406  004737  034240                    JSR     PC,LODSEC       ;LOAD BUF WITH TRACK + 100(OCT)
6218  024412  112765  000123  000001            MOVB    #WRDATA,P.CMND(R5) ;SET WRITE DATA COMMAND
6219  024420  004737  040252                    JSR     PC,DRVCAL       ;RE-WRITE A SECTOR
6220  024424  004737  034262                    JSR     PC,TRKCHK       ;WRITE-CHECK ALL 3 SECTORS
6221  024430  105265  000005                    INCB    P.TRCK(R5)      ;INCR THE TRACK NO.
6222  024434  122765  000003  000005            CMPB    #3,P.TRCK(R5)   ;SEE IF ALL 3 TRACKS RE-WRITTEN YET
6223  024442  001355                            BNE     6$              ;BR IF NOT DONE YET
6224
6225
6226
6227  024444                            RWDTST:
6228          000040                    RWTINX=<$TN-1>*2                ;R/W TEST INDEX
6229                                    ;;*****************************************************************
6230                                    ;*TEST 21       READ/WRITE DATA TEST
6231                                    ;*THE READ/WRITE DATA TEST HAS 2 DIFFERENT VERSIONS, DEPENDING
6232                                    ;*ON THE VALUE OF THE DATA PATTERN PARAMETER (PT).
6233                                    ;*
6234                                    ;*  FOR PT = 0 :
```

```
                                ;*THIS TEST I> 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.
                                ;;*****************************************************************
024444  000004                  TST21:  SCOPE
024446  012737  000021  001324          MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
024454  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
024460  004737  030464                  JSR     PC,CHKITR       ;SEE IF ITER. NO. = 0 FOR THIS TEST
024464  000137  025356                  JMP     $EOP            ;JUMP TO END-OF-PASS ROUTINE
                                ;RETURN HERE FROM CHKITR IF TEST SHOULD BE RUN
024470  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER PARAMS AND SUB-SYS
024474  004737  025766                  JSR     PC,PREPKB       ;PREPARE FOR POSSIBLE KBD INPUT
024500  105037  003134                  CLRB    XOVLAD          ;INIT XXDP OVERLAID INDICATOR
024504  032777  000040  154426          BIT     #BIT5,@SWR      ;SEE IF WRITES INHIBITED
024512  001002                          BNE     2$              ;IF YES,DON'T CHANGE RAND DATA PAT
024514  004737  034346                  JSR     PC,LOC?14       ;GENERATE PSEUDO-RAND PAT 14
024520  013737  005760  005544  2$:     MOV     PT,PATRN        ;GET A COPY OF PT
024526  005737  005760                  TST     PT              ;SEE IF QUICK VERIFY DATA TEST DESIRED
024532  001020                          BNE     7$              ;BR IF NOT QUICK VERIFY
                                ;:ET PARAMETERS FOR QUICK VERIFY DEFAULT DATA TEST
024534  012765  063520  000010          MOV     #RWBUF,P.BALO(R5) ;SET R/W BUFFER ADDRESS
024542  005065  000002                  CLR     P.CYLN(R5)      ;INIT CYL TO 0
024546  105065  000005          4$:     CLRB    P.TRCK(R5)      ;INIT TRACK TO 0
024552  012765  175000  000012  5$:     MOV     #-1536.,P.WC(R5) ;SET WORD COUNT FOR 6 SECTORS
024560  105065  000004                  CLRB    P.SECT(R5)      ;INIT SECTOR TO 0
024564  004737  035716          6$:     JSR     PC,CHKLIM       ;CHK WRD COUNT AND PACK ADR TO AVOID OVERFLOW
024570  025332                          30$                     ;RETURN ADDR FOR TRANSFER NOT ALLOWED
024572  000457                          BR      18$             ;PROCEED
                                ;SET PARAMETERS FOR NON-DEFAULT DATA TEST
024574  013765  005762  000010  7$:     MOV     MA,P.BALO(R5)   ;SET BA BITS 0-15
024602  013700  005764                  MOV     MA+2,R0         ;GET MA BITS 16-21
024606  042700  177774                  BIC     #177774,R0      ;MASK FOR LO 2 BITS
024612  150065  000007                  BISB    R0,P.BAHI(R5)   ;SET BA BITS 16,17
024616  013737  005762  005602          MOV     MA,PMA          ;SET BUFFER ADDRESS
024624  013737  005764  005604          MOV     MA+2,PMA+2
```

```
6291  024632  105237  003134                  INCB    XOVLAD          ;SET XXDP OVERLAID INDICATOR
6292  024636  005737  055662                  TST     $KT11           ;SEE IF MEMORY MANAGEMENT PRESENT
6293  024642  100002                          BPL     9$              ;BR IF NO MEM. MGT.
6294                                ;PREPARE PAGE ADDRESS REGISTERS FOR RELOCATION
6295  024644  004737  026532                  JSR     PC,PREPAR       ;PREPARE MEM MGT FOR RELOCATION
6296                                ;INITIALIZE TEST PARAMETERS
6297  024650  013765  005732  000002  9$:     MOV     FC,P.CYLN(R5)   ;INIT CYL TO FC
6298  024656  012701  000001        44$:     MOV     #1,R1           ;INIT PATTERN BIT POINTER
6299  024662  030137  005760        10$:     BIT     R1,PT           ;SEE IF THIS PATTERN SELECTED
6300  024666  001003                          BNE     14$             ;BR IF THIS PATTERN IS SELECTED
6301  024670  006301                11$:     ASL     R1              ;SHIFT PATTERN BIT POINTER
6302  024672  001571                          BEQ     25$             ;BR IF ALL PATTERNS CHECKED ON THIS CYLINDER
6303  024674  000772                          BR      10$             ;GO CHECK ANOTHER PATTERN
6304  024676  113765  005740  000005  14$:     MOVB    FT,P.TRCK(R5)   ;INIT TRACK TO FT
6305  024704  113765  005516  000004  16$:     MOVB    FS,P.SECT(R5)   ;INITIALIZE SECTOR TO FS
6306  024712  013765  005766  000012  17$:     MOV     WC,P.WC(R5)     ;SET WORD COUNT FOR WC WORDS
6307  024720  005465  000012                  NEG     P.WC(R5)
6308  024724  004737  035716                  JSR     PC,CHKLIM       ;CHECK WRD CNT AND PACK ADR TO AVOID OVERFLOW
6309  024730  025256                          25$                     ;RETURN ADDRESS FOR TRANSFER NOT ALLOWED
6310                                ;PERFORM TESTS AT CURRENT ADDRESS
6311  024732  012737  024732  001110  18$:     MOV     #.,$LPERR       ;SET NEW LOOP ON ERROR ADRS
6312  024740  004737  030416                  JSR     PC,SETTUP       ;SET UP FOR LOOP ON ERROR
6313  024744  004737  037562                  JSR     PC,SVPRMS       ;STORE PARAMS FOR THIS XFER
6314  024750  004737  034562                  JSR     PC,LODBUF       ;LOAD DATA INTO R/W BUFFER
6315  024754  105037  003140                  CLRB    REISSU          ;DUAL-ACC COMMAND RE-ISSUE FLAG
6316  024760  032777  000040  154152          BIT     #BIT5,@SWR      ;SEE IF DATA WRITES INHIBITED
6317  024766  001005                          BNE     20$             ;BR IF INHIBITED
6318  024770  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
6319  024776  004737  037674                  JSR     PC,TRNSFR       ;WRITE THE DATA
6320  025002  032777  000020  154130  20$:     BIT     #BIT4,@SWR      ;SEE IF WRITE CHECKS INHIBITED
6321  025010  001014                          BNE     22$             ;BR IF INHIBITED
6322  025012  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
6323  025020  004737  037674                  JSR     PC,TRNSFR       ;DO WRITE-CHECK OF DATA WRITTEN
6324  025024  105737  003141                  TSTB    WCEFLG          ;SEE IF A WCE ERROR OCCURRED
6325  025030  001404                          BEQ     22$             ;BR IF NOT
6326  025032  032777  000002  154100          BIT     #BIT1,@SWR      ;SEE IF READ AND COMPARE SHOULD BE DONE
6327  025040  001417                          BEQ     24$             ;BR IF NOT
6328  025042  032777  000010  154070  22$:     BIT     #BIT3,@SWR      ;SEE IF READ AND COMPARE INHIBITED
6329  025050  001013                          BNE     24$             ;BR IF INHIBITED
6330  025052  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
6331  025060  004737  037674                  JSR     PC,TRNSFR       ;READ THE DATA
6332  025064  032777  000004  154046          BIT     #BIT2,@SWR      ;SEE IF SOFTWARE COMPARES INHIBITED
6333  025072  001002                          BNE     24$             ;BR IF INHIBITED
6334  025074  004737  035010                  JSR     PC,CMPBUF       ;PERFORM SOFTWARE COMPARE OF DATA
6335  025100  105737  003136          24$:     TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
6336  025104  001405                          BEQ     27$             ;BR IF NOT DUAL-ACCESS
6337  025106  112765  000140  000001          MOVB    #RELEAS,P.CMND(R5) ;SET DRIVE RELEASE COMMAND
6338  025114  004737  040252                  JSR     PC,DRVCAL       ;RELEASE THE DRIVE
6339  025120  104411                27$:     SCOPER                  ;CHECK FOR INTERNAL LOOP ON ERROR
6340  025122  005737  005760                  TST     PT              ;SEE IF QUICK VERIFY DATA TEST DESIRED
6341  025126  001031                          BNE     40$             ;BR IF NOT QUICK VERIFY
6342                                ;INCREMENT PACK ADDRESS FOR QUICK VERIFY DEFAULT DATA TEST
6343  025130  062765  000006  000004          ADD     #6,P.SECT(R5)   ;INCREMENT SECTOR BY 6
6344  025136  126527  000004  000030          CMPB    P.SECT(R5),#24. ;SEE IF SECTOR LIMIT REACHED YET
6345  025144  002002                          BGE     34$             ;BR IF LIMIT EXCEEDED
6346  025146  000137  024564                  JMP     6$              ;GO BACK AND TEST
```

```
 6347  025152  105265  000005          34$:    INCB    P.TRCK(R5)        ;INCREMENT TRACK
 6348  025156  122765  000003  000005          CMPB    #3,P.TRCK(R5)     ;SEE IF TRACK LIMIT EXCEEDED
 6349  025164  001402                          BEQ     36$               ;BR IF YES
 6350  025166  000137  024552                  JMP     5$
 6351  025172  005265  000002          36$:    INC     P.CYLN(R5)        ;INCREMENT CYL
 6352  025176  023765  017710  000002          CMP     LCP1,P.CYLN(R5)   ;SEE IF CYL LIMIT EXCEEDED
 6353  025204  001452                          BEQ     30$               ;BR IF YES
 6354  025206  000137  024546                  JMP     4$
 6355                                  ;INCREMENT PACK ADDRESS FOR NON-DEFAULT DATA TEST
 6356  025212  063765  005756  000004  40$:    ADD     IS,P.SECT(R5)     ;INCREMENT SECTOR BY IS
 6357  025220  126537  000004  005520          CMPB    P.SECT(R5),LS     ;SEE IF LS EXCEEDED
 6358  025226  003631                          BLE     17$               ;BR IF NOT YET
 6359  025230  116503  000005                  MOVB    P.TRCK(R5),R3     ;GET TRACK NO.
 6360  025234  063703  005744                  ADD     IT,R3             ;INCREMENT TRACK
 6361  025240  110365  000005                  MOVB    R3,P.TRCK(R5)     ;PUT IT BACK
 6362  025244  020337  005742                  CMP     R3,LT             ;SEE IF LT EXCEEDED
 6363  025250  003615                          BLE     16$               ;BR IF NOT YET
 6364  025252  000137  024670                  JMP     11$               ;JMP TO CHECK FOR NEXT PATTERN
 6365  025256  023737  005732  005734  25$:    CMP     FC,LC             ;SEE IF FC>LC
 6366  025264  003011                          BGT     26$               ;BR IF FC>LC
 6367  025266  063765  005736  000002          ADD     IC,P.CYLN(R5)     ;INCREMENT THE CYLINDER
 6368  025274  026537  000002  005734          CMP     P.CYLN(R5),LC     ;SEE IF LC EXCEEDED
 6369  025302  003013                          BGT     30$               ;BR IF LC EXCEEDED
 6370  025304  000137  024656                  JMP     44$
 6371  025310  163765  005736  000002  26$:    SUB     IC,P.CYLN(R5)     ;DECREMENT THE CYLINDER
 6372  025316  026537  000002  005734          CMP     P.CYLN(R5),LC     ;SEE IF LC EXCEEDED
 6373  025324  002402                          BLT     30$               ;BR IF LC EXCEEDED
 6374  025326  000137  024656                  JMP     44$
 6375  025332  105737  003134          30$:    TSTB    XOVLAD            ;SEE IF XXDP MIGHT BE OVERLAID NOW
 6376  025336  001402                          BEQ     31$               ;BR IF NOT
 6377  025340  004737  027210                  JSR     PC,GETXDP         ;RESTORE XXDP LOADER, IF SAVED
 6378  025344  105737  003144          31$:    TSTB    UBMPRS            ;SEE IF UNIBUS MAP PRESENT
 6379  025350  001402                          BEQ     32$               ;BR IF NOT
 6380  025352  005037  172516                  CLR     @#SR3             ;DISABLE 22-BIT MODE AND UNIBUS MAP
 6381  025356                          32$:
 6382
 6383
 6384
 6385                                  .SBTTL  END OF PASS ROUTINE
 6386
 6387                                  ;;**********************************************************************
 6388                                  ;*INCREMENT THE PASS NUMBER ($PASS)
 6389                                  ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
 6390                                  ;*IF THERES A MONITOR GO TO IT
 6391                                  ;*IF THERE ISN'T JUMP TO APTPAS
 6392
 6393  025356                          $EOP:
 6394  025356  000004                          SCOPE
 6395  025360  005237  001330                  INC     $DEVCT            ;INCREMENT DEVICE COUNT FOR APT
 6396  025364  000137  016754                  JMP     NEWDRV            ;GO SEE IF MORE DRIVES TO TEST ON THIS PASS
 6397  025370                          DUNPAS: ;THIS IS TRULY THE END OF A PASS
 6398  025370  042777  000100  153546          BIC     #BIT6,@$TKS       ;DISABLE TTY KBD INTERRUPT
 6399  025376  005037  001102                  CLR     $TSTNM            ;;ZERO THE TEST NUMBER
 6400  025402  005037  001304                  CLR     $TIMES            ;;ZERO THE NUMBER OF ITERATIONS
 6401  025406  005237  001326                  INC     $PASS             ;;INCREMENT THE PASS NUMBER
 6402  025412  042737  100000  001326          BIC     #100000,$PASS     ;;DON'T ALLOW A NEG. NUMBER
```

```
 6403  025420  005327                        DEC     (PC)+           ;;LOOP?
 6404  025422  000001                $EOPCT: .WORD   1
 6405  025424  003022                        BGT     $DOAGN          ;;YES
 6406  025426  012737                        MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 6407  025430  000001                $ENDCT: .WORD   1
 6408  025432  025422                        $EOPCT
 6409  025434  104401  025501              TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 6410  025440  013746  001326              MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 6411  025444  104405                      TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 6412  025446  104401  025476              TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 6413  025452  013700  000042      $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 6414  025456  001405                      BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 6415  025460  000005                      RESET                   ;;CLEAR THE WORLD
 6416  025462  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 6417  025464  000240                      NOP                     ;;SAVE ROOM
 6418  025466  000240                      NOP                     ;;FOR
 6419  025470  000240                      NOP                     ;;ACT11
 6420  025472                        $DOAGN:
 6421  025472  000137                      JMP     @(PC)+          ;;RETURN
 6422  025474  025516                $RTNAD: .WORD   APTPAS
 6423  025476     377     377     000 $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 6424  025501     015  042412  042116 $ENDMG: .ASCIZ  <15><12>/END PASS #/
 6425  025506  050040  051501  020123
 6426  025514  000043
 6427  025516  122737  000001  001340 APTPAS: CMPB    #APTENV,$ENV    ;RUN UNDER THE APT ?
 6428  025524  001007                      BNE     2$              ;BRANCH IF NOT
 6429  025526  023727  001326  000002        CMP     $PASS,#2        ;TWO PASSES ?
 6430  025534  103403                      BLO     2$              ;BRANCH IF NOT
 6431  025536  005237  001102          1$: INC     $TSTNM          ;INCREMENT THE TEST NUMBER
 6432  025542  000775                      BR      1$               LOOPING FOR APT TO DUMP NEXT PROG
 6433  025544  000137  016742          2$: JMP     NEWPAS          ;JMP TO MAIN LOOP
 6434
 6435
 6436
 6437                                  ;THIS IS THE ABNORMAL RETURN FROM THE DRIVER,
 6438                                  ;  WHICH IS USED WHEN NED INDICATION IS EXPECTED (AND VALID).
 6439                                  ;    (NED = NON-EXISTENT DRIVE)
 6440  025550  032765  010000  000020 NEDHDL: BIT     #NED,P.CS2(R5)  ;SEE IF NED ON DRIVE SELECT
 6441  025556  001002                      BNE     1$              ;BR IF NED SET
 6442  025560  000137  042130              JMP     ERRHDL          ;GO HANDLE  OTHER ERROR
 6443  025564  010046                  1$: MOV     R0,-(SP)        ;SAVE R0,R1
 6444  025566  010146                      MOV     R1,-(SP)
 6445  025570  012700  000001              MOV     #1,R0           ;SET BIT POINTER
 6446  025574  005001                      CLR     R1              ;CLEAR COUNTER
 6447  025576  120137  005510          2$: CMPB    R1,DRIVE        ;SEE IF R1 = CURRENT DRIVE NUMBER
 6448  025602  001403                      BEQ     3$              ;BR IF  =
 6449  025604  005201                      INC     R1              ;INCREMENT COUNTER
 6450  025606  006300                      ASL     R0              ;SHIFT BIT POINTER
 6451  025610  000772                      BR      2$              ;TRY AGAIN
 6452  025612  040037  005540          3$: BIC     R0,NEWON        ;CLEAR ONLINE BIT FOR  THIS DRIVE
 6453  025616  012601                      MOV     (SP)+,R1        ;RESTORE R0,R1
 6454  025620  012600                      MOV     (SP)+,R0
 6455  025622  000137  044306              JMP     RETNML          ;TAKE  NORMAL  RETURN
 6456
 6457
 6458                                  ;;**************************************************************
```

```
                        .SBTTL  KBDHDL - TTY KEYBOARD INTERRUPT HANDLER
                        ;;*****************************************************************
025626  010146          KBDHDL: MOV     R1,-(SP)        ;SAVE R1 ON STACK
025630  017701  153312          MOV     @$TKB,R1        ;READ A CHAR FROM KBD BUFFER
025634  042701  177600          BIC     #177600,R1      ;MASK OUT UNUSED BITS
025640  120127  000172          CMPB    R1,#172         ;SEE IF LOWER CASE TYPED
025644  003005                  BGT     20$             ;BR IF NOT
025646  120127  000141          CMPB    R1,#141
025652  002402                  BLT     20$             ;BR IF NOT
025654  042701  000040          BIC     #BIT5,R1        ;MAKE IT UPPER CASE
025660  010137  005532  20$:     MOV     R1,INTCHR       ;SAVE INPUT CHARACTER
025664  122701  000003          CMPB    #003,R1         ;SEE IF (^C) TYPED
025670  001007                  BNE     2$              ;BR IF NOT (^C)
025672  104401  012353          TYPE    ,CNTRLC         ;ECHO (^C)
025676  042777  000100  153240  1$:     BIC     #BIT6,@$TKS     ;CLEAR KBD INTERRUPT ENABLE BIT
025704  012601                  MOV     (SP)+,R1        ;RESTORE R1
025706  000002                  RTI                     ;RETURN
025710  122701  000032          2$:     CMPB    #032,R1         ;SEE IF (^Z) TYPED
025714  001003                  BNE     3$              ;BR IF NOT (^Z)
025716  104401  012360          TYPE    ,CNTRLZ         ;ECHO (^Z)
025722  000765                  BR      1$              ;RETURN
025724  122701  000022          3$:     CMPB    #022,R1         ;SEE IF (^R) TYPED
025730  001003                  BNE     4$              ;BR IF NOT (^R)
025732  104401  012365          TYPE    ,CNTRLR         ;ECHO (^R)
025736  000757                  BR      1$              ;RETURN
025740  122701  000007          4$:     CMPB    #007,R1         ;SEE IF (^G) TYPED
025744  001003                  BNE     5$              ;BR IF NOT (^G)
025746  104401  012377          TYPE    ,CNTRLG         ;ECHO (^G)
025752  000751                  BR      1$              ;RETURN
025754  104401  005532          5$:     TYPE    ,INTCHR         ;ECHO INPUT
025760  104401  001315          TYPE    ,$CRLF          ;DO <CR> AND <LF>
025764  000744                  BR      1$              ;RETURN


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

025766  005077  153154  PREPKB: CLR     @$TKB           ;CLEAR KBD BUFFER AND DONE BIT
025772  005037  005532          CLR     INTCHR          ;CLEAR TTY INPUT WORD
025776  052777  000100  153140  BIS     #BIT6,@$TKS     ;ENABLE KBD INTERRUPT
026004  000207                  RTS     PC              ;RETURN


                        ;;*****************************************************************
                        .SBTTL  ECOBAD - ECHO BAD TTY INPUT
                        ;*THIS SUBROUTINE ECHOS A CHARACTER WHICH HAD BEEN
```

```
                        ;*TYPED IN AND WAS DETERMINED TO BE INVALID FOR SOME
                        ;*REASON, FOLLOWED BY A QUESTION MARK (?). THEN, A <CR>
                        ;*AND <LF> ARE DONE.
                        ;* CALL -     JSR      PC,ECOBAD
                        ;;*****************************************************************
026006  104401  005532 ECOBAD: TYPE    ,INTCHR          ;ECHO BAD CHARACTER
026012  104401  001314         TYPE    ,$QUES           ;TYPE <?> AND <CR>, <LF>
026016  000207                 RTS     PC               ;RETURN


                        ;;*****************************************************************
                        ;*SIZMEM - SIZE MEMORY, SET LIMITS
                        ;*THIS SUBROUTINE SIZES MEMORY BY CALLING SUBROUTINE $SIZE, AND
                        ;*IT USES THE VALUES RETURNED, TO SET THE LIMITS ON PARAMETER MA.
                        ;*IT ALSO TYPES "LAST PHYS MEM ADR = XXXXXXXX" (LEAD ZEROS SUPRS'D),
                        ;*AND SETS THE FLAG UBMPRS = 1 IF 22-BIT ADDRESSES ARE USED (11/70).
                        ;;*****************************************************************
026020  104407         SIZMEM: SAVREG                   ;SAVE R0-R5
026022  012737  000200  055662         MOV     #200,$KT11       ;SET MEM MGT KEY FOR $SIZE
026030  004737  055624         JSR     PC,$SIZE         ;SIZE MEMORY
026034  005737  055662         TST     $KT11            ;SEE IF MEM MGT PRESENT
026040  100406                 BMI     8$               ;BR IF MEM MGT PRESENT
026042  013737  056126  006126         MOV     $LSTAD,MAHILM    ;SET MEM LIMIT LO BITS
026050  005037  006130         CLR     MAHILM+2         ;CLEAR MEM LIMIT HI BITS
026054  000436                 BR      16$              ;GO TYPE LAST ADDRESS
                        ;SHIFT SAF LEFT 6, AND PUT IN R1-R0
026056  013700  056130 8$:     MOV     $LSTBK,R0        ;LO BITS
026062  005001                 CLR     R1               ;HI BITS
026064  012702  000006         MOV     #6,R2            ;SET LOOP COUNT = 6
026070  000241         12$:    CLC                      ;ROTATE LOOP
026072  006100                 ROL     R0
026074  006101                 ROL     R1
026076  005302                 DEC     R2
026100  001373                 BNE     12$
                        ;ADD VIRTUAL ADDRESS TO SHIFTED SAF TO GET PHYSICAL ADDRESS
026102  063700  056126         ADD     $LSTAD,R0        ;ADD LO BITS
026106  005501                 ADC     R1               ;HI BITS
026110  010037  006126         MOV     R0,MAHILM        ;SET LO BITS OF MEM LIMIT
026114  010137  006130         MOV     R1,MAHILM+2      ;SET HI BITS OF MEM LIMIT
026120  000241                 CLC
026122  006100                 ROL     R0
026124  006101                 ROL     R1
026126  006100                 ROL     R0
026130  006101                 ROL     R1
026132  006100                 ROL     R0
026134  006101                 ROL     R1
026136  020127  000037         CMP     R1,#37           ;SEE IF 22 BIT ADDR
026142  103403                 BLO     16$              ;BR IF NO
026144  112737  000001  003144         MOVB    #1,UBMPRS        ;SET "UNIBUS MAP PRESENT" FLAG
                        ;TYPE "LAST PHYS MEM ADR = XXXXXXXX"
026152  104401  007354 16$:     TYPE    ,LSTMEM          ;TYPE "LAST PHYS MEM ADR ="
026156  012746  006126         MOV     #MAHILM,-(SP)    ;PUT POINTER ON STACK
026162  004737  053366         JSR     PC,@#$DB2O       ;CONVERT BINARY TO OCTAL ASCII
026166  004737  053702         JSR     PC,@#$SUPRS      ;TYPE "XXXXXXXX"
```

```
                                TYPE    ,$CRLF          ;TYPE <CR>,<LF>
                                TYPE    ,$CRLF
                                RESREG                  ;RESTORE R0-R5
                                RTS     PC              ;RETURN


                        ;;*******************************************************************
                        ;* GETPRM - INPUT A SIX-DIGIT OCTAL NO. ON TTY KBD
                        ;*ENTER SUBROUTINE WITH OLD PARAMETER VALUE ON STACK. SUBROUTINE
                        ;*TYPES OLD VALUE, AND INPUTS NEW VALUE, RETURNING NEW VALUE ON
                        ;*TOP OF STACK.
                        ;;*******************************************************************
026206  016646  000002  GETPRM: MOV     2(SP),-(SP)     ;GET OLD VALUE
026212  104403                  TYPOS                   ;TYPE IT
026214     006                  .BYTE   6               ;SIX DIGITS
026215     000                  .BYTE   0               ;SUPPRESS LEADING ZEROS
026216  104401  007564          TYPE    ,NEWMSG         ;TYPE "  NEW = "
026222  004737  030560          JSR     PC,RDCHRS       ;READ NEW VALUE FROM KBD
026226  026260                  7$                      ;(^C) RETURN ADDRESS
026230  026260                  7$                      ;(^Z) RETURN ADDRESS
026232  026260                  7$                      ;(^U) OR ERROR RETURN ADDRESS
026234  005700                  TST     R0              ;SEE IF ANY CHARS TYPED
026236  001001                  BNE     4$              ;BR IF YES
026240  000207                  RTS     PC              ;RETURN - OLD VALUE UNCHANGED
026242  020027  000006  4$:     CMP     R0,#6           ;SEE IF > 6 CHARS TYPED
026246  003407                  BLE     8$              ;BR IF NOT BAD
026250  104401  005272  6$:     TYPE    ,BUFFO          ;ECHO BAD INPUT
026254  104401  001314          TYPE    ,$QUES
026260  162716  000010  7$:     SUB     #10,(SP)        ;FIX ERROR RETURN PC
026264  000207                  RTS     PC              ;ERROR RETURN
026266  012746  005272  8$:     MOV     #BUFFO,-(SP)    ;PUT POINTER TO CHARS ON STACK
026272  004737  052430          JSR     PC,OCTBIN       ;CONVERT DIGITS TO BINARY
026276  026250                  6$                      ;ERROR RETURN ADDRESS
026300  012600                  MOV     (SP)+,R0        ;GET NEW BINARY VALUE
026302  005737  052562          TST     $HIOCT          ;SEE IF HI BITS ARE 0
026306  001360                  BNE     6$              ;BR IF NOT
026310  010066  000002          MOV     R0,2(SP)        ;PUT NEW VALUE ON STACK
026314  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.
                        ;;*******************************************************************
026316  022737  000176  001140  GTSWRG: CMP     #SWREG,SWR      ;SEE IF SOFTWARE SWR SELECTED
026324  001010                  BNE     6$              ;BR IF NOT
026326  013746  000176          MOV     SWREG,-(SP)     ;PUT OLD VALUE ON STACK
026332  104401  007555          TYPE    ,SWRMSG         ;TYPE "SWR = "
026336  004737  026206          JSR     PC,GETPRM       ;TYPE OLD, GET NEW SWREG VALUE
026342  012637  000176          MOV     (SP)+,SWREG     ;STORE NEW VALUE
026346  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.

                        ;;*****************************************************************
026350  104407          INITMM: SAVREG                  ;SAVE R0-R5
026352  005001                  CLR     R1              ;INIT FOR PAR LOADING
026354  012702  172340          MOV     #KIPAR0,R2      ;ADDR OF FIRST PAR
026360  012703  000010          MOV     #8.,R3          ;LOAD 8 PAR'S AND 8 PDR'S
026364  012762  077406  177740  4$:     MOV     #77406,-40(R2)  ;PDR = 4K,UP,READ/WRITE
026372  010122                  MOV     R1,(R2)+        ;LOAD A PAR
026374  062701  000200          ADD     #200,R1         ;UPDATE FOR NEXT PAR
026400  005303                  DEC     R3              ;DECREMENT LOOP COUNTER
026402  001370                  BNE     4$              ;LOOP UNTIL ALL 8 ARE LOADED
026404  012742  177600          MOV     #177600,-(R2)   ;SET UP KIPAR7 FOR I/O PAGE
026410  105737  003144          TSTB    UBMPRS          ;SEE IF 22-BIT ADDRESSES
026414  001403                  BEQ     10$             ;BR IF NOT
026416  012737  000060  172516  MOV     #60,@#SR3       ;ENABLE 22-BIT MODE,AND UNIBUS MAP
026424  012737  001000  177572  10$:    MOV     #BIT9,@#SR0     ;ENABLE KT11 TRAPS
026432  104410                  RESREG                  ;RESTORE R0-R5
026434  000207                  RTS     PC              ;RETURN


                        ;;*****************************************************************
                        ;*SERVICE ROUTINE FOR MEM MGT TRAPS
                        ;;*****************************************************************
026436  004737  041650  KTERHD: JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
026442  013737  177572  001174          MOV     @#SR0,$REG5     ;GET SR0
026450  013737  177574  001176          MOV     @#SR1,$REG6     ;GET SR1
026456  013737  177576  001200          MOV     @#SR2,$REG7     ;GET SR2
026464  012737  026510  000004          MOV     #8$,@#ERRVEC    ;SET TIME-OUT VECTOR
026472  012737  000340  000006          MOV     #PR7,@#ERRVEC+2
026500  013737  172516  001202          MOV     @#SR3,$REG10    ;GET SR3
026506  000406                  BR      10$
026510  022626          8$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
026512  112737  000003  063472          MOVB    #3,DF30+18.     ;FIX PRINTOUT FOR NO SR3
026520  105037  062145                  CLRB    DH704+11.
026524  104121          10$:    ERROR   121             ;KT11 FAILURE
026526  000137  044100                  JMP     HLTPRG          ;ABORT !!!


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

```
                                ;;*****************************************************************
026532  104407                  PREPAR: SAVREG                  ;SAVE R0-R5
026534  004737  026350                  JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
026540  013700  005602                  MOV     PMA,R0          ;LO BITS OF MA
026544  042700  017777                  BIC     #17777,R0       ;MASK FOR BITS 13-15
026550  013701  005604                  MOV     PMA+2,R1        ;HI BITS OF MA
026554  010003                          MOV     R0,R3           ;SAVE THESE BITS
026556  010104                          MOV     R1,R4
026560  006100                          ROL     R0              ;GET MA BITS 13-21 INTO R1 BITS 7-15
026562  006101                          ROL     R1
026564  006100                          ROL     R0
026566  006101                          ROL     R1
026570  000301                          SWAB    R1
026572  006100                          ROL     R0
026574  106001                          RORB    R1
026576  010137  003206                  MOV     R1,SAVPAR       ;CONSTANT FOR LOADING PAR6 LATER
026602  105737  003144                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP ENABLED
026606  001420                          BEQ     9$              ;BR IF NOT (NO UNIBUS MAPPING)
                                ;PREPARE FOR 22-BIT ADDRESSING -  LOAD UNIBUS MAP REGISTERS
026610  012700  170200                  MOV     #MAPLOO,R0      ;STARTING ADDR OF MAP REGISTERS
026614  012701  000037                  MOV     #31.,R1         ;SET REGISTER COUNTER
026620  010320                  4$:     MOV     R3,(R0)+        ;LOAD A MAP REGISTER
026622  010420                          MOV     R4,(R0)+
026624  062703  020000                  ADD     #20000,R3       ;ADD 4K WORDS
026630  005504                          ADC     R4
026632  077106                          SOB     R1,4$           ;LOOP UNTIL 31(DEC) ARE LOADED
026634  042765  160000  000010        BIC     #160000,P.BALO(R5) ;CLEAR BA BITS 13-17
026642  142765  000003  000007        BICB    #3,P.BAHI(R5)
026650  104410                  9$:     RESREG                  ;RESTORE R0-R5
026652  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;* FNDXDP - FIND STARTING ADR OF XXDP LOADER , AND STORE IT
                                ;*IN XXDPAD.
                                ;;*****************************************************************
026654  005737  006130          FNDXDP: TST     MAHILM+2        ;TEST HI BITS OF UPPER MEMORY LIMIT
026660  001404                          BEQ     6$              ;BR IF HI BITS ARE 0
026662  012737  160000  005546  4$:     MOV     #160000,XXDPAD  ;START OF 28K IS END OF XXDP
026670  000412                          BR      8$
026672  023727  006126  157776  6$:     CMP     MAHILM,#157776  ;SEE IF MEM LIMIT -> OR = 157776
026700  103370                          BHIS    4$              ;BR IF YES
026702  013737  006126  005546          MOV     MAHILM,XXDPAD   ;HI MEM LIMIT IS END OF XXDP
026710  062737  000002  005546          ADD     #2,XXDPAD       ;ADD 2 BYTES
026716  162737  006000  005546  8$:     SUB     #6000,XXDPAD    ;COMPUTE START OF XXDP
026724  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*CHKXDP - CHECK FOR IMMINENT OVERLAY OF XXDP LOADER
                                ;*THIS SUBROUTINE DETERMINES IF THE CURRENT MA AND WC COMBINATION
                                ;*WOULD CAUSE OVERLAY OF THE XXDP LOADER ON A WRITE TRANSFER.
                                ;*IF SO, AN ATTEMPT IS MADE TO MOVE THE LOADER INTO UNUSED MEMORY.
                                ;*IF THIS IS NOT POSSIBLE, A RETURN IS MADE TO THE ADDRESS
```

```
                                ;*FOLLOWING THE CALL. THE SAVE ADDRESS IS STORED IN XXDPSAV AND XXDPSAV+2.
                                ;*IF IT IS NOT NECESSARY TO MOVE THE LOADER, A NORMAL RETURN IS MADE.
                                ;*  CALL  -     JSR     PC,CHKXDP
                                ;*              <RETURN ADR FOR NO SAVE>
                                ;;*************************************************************
026726  104407                  CHKXDP: SAVREG                  ;SAVE R0-R5
                                ;SEE IF LOADER MUST BE PROTECTED
026730  105737  003136                  TSTB    DULACS          ;SEE IF DUAL-ACCESS DATA TEST
026734  001005                  BNE     2$              ;BR IF YES
026736  012700  005626          MOV     #TSTLST,R0      ;GET ADRS OF TEST LIST
026742  005760  000040          TST     RWTINX(R0)      ;SEE IF R/W DATA TEST TO BE RUN
026746  001406                  BEQ     4$              ;EXIT IF NOT TO BE RUN
026750  005737  005760  2$:     TST     PT              ;SEE IF DEFAULT DATA TEST
026754  001403                  BEQ     4$              ;BR IF DEFAULT DATA TEST
026756  005737  005764          TST     MA+2            ;SEE IF HI MEM ADR = 0
026762  001404                  BEQ     6$              ;BR IF 0
026764  062716  000002  4$:     ADD     #2,(SP)         ;FIX UP NORMAL RETURN PC
026770  104410          5$:     RESREG                  ;RESTORE R0-R5
026772  000207                  RTS     PC              ;RETURN
026774  013700  005766  6$:     MOV     WC,R0           ;GET WORD COUNT LO BITS
027000  005300                  DEC     R0              ;DECREMENT IT
027002  005001                  CLR     R1              ;SET HI BITS = 0
027004  000241                  CLC                     ;DOUBLE IT FOR BYTES
027006  006100                  ROL     R0
027010  006101                  ROL     R1
027012  063700  005762          ADD     MA,R0           ;COMPUTE LAST ADR OF XFER IN R1-R0
027016  005501                  ADC     R1
027020  063701  005764          ADD     MA+2,R1
027024  105037  003135          CLRB    XDPSVD          ;INIT LOADER SAVE INDICATOR
027030  004737  026654          JSR     PC,FNDXDP       ;COMPUTE START ADR OF XXDP
027034  013703  005546          MOV     XXDPAD,R3       ;GET START OF XXDP LOADER
027040  062703  005776          ADD     #5776,R3        ;COMPUTE LAST ADDRESS OF XXDP
027044  023703  005762          CMP     MA,R3           ;SEE IF MA > LAST ADR OF XXDP
027050  101345                  BHI     4$              ;BR IF YES - DON'T MOVE XXDP
027052  005701                  TST     R1              ;CHECK HI BITS OF LAST XFER ADDRESS
027054  001003                  BNE     8$              ;BR IF NOT 0
027056  020037  005546          CMP     R0,XXDPAD       ;CHECK LO BITS OF LAST XFER ADR
027062  103740                  BLO     4$              ;BR IF < XXDP ADR - DON'T MOVE XXDP
027064  105237  003135  8$:     INCB    XDPSVD          ;SET SAVE INDICATOR
                                ;ATTEMPT TO FIND A SAVE AREA FOR LOADER
027070  023727  005762  100000  CMP     MA,#100000      ;SEE IF MA > OR = 16K
027076  103410                  BLO     12$             ;BR IF NOT
027100  012737  072000  005550  MOV     #72000,XDPSAV   ;SAVE ADR = 72000
027106  005037  005552          CLR     XDPSAV+2
027112  004737  027202  10$:     JSR     PC,SAVXDP       ;SAVE THE LOADER
027116  000722                  BR      4$              ;BR TO RETURN
027120  062700  000002  12$:     ADD     #2,R0           ;ADD 2 BYTES TO LAST ADR OF XFER
027124  005501                  ADC     R1
027126  010002                  MOV     R0,R2           ;GET A COPY
027130  010103                  MOV     R1,R3
027132  062702  005776          ADD     #5776,R2        ;COMPUTE END OF SAVE AREA
027136  005503                  ADC     R3
027140  020337  006130          CMP     R3,MAHILM+2     ;SEE IF THIS EXCEEDS MEMORY LIMIT
027144  101011                  BHI     16$             ;BR IF YES - CAN'T SAVE LOADER
027146  001003                  BNE     14$             ;BR IF < MEM LIMIT
027150  020237  006126          CMP     R2,MAHILM       ;SEE IF LO BITS EXCEED MEM LIMIT
```

```
6795  027154  101005                  BHI     16$             ;BR IF YES - CAN'T SAVE LOADER
6796  027156  010037  005550  14$:    MOV     R0,XDPSAV       ;SET SAVE ADR IN HI MEMORY
6797  027162  010137  005552          MOV     R1,XDPSAV+2
6798  027166  000751                  BR      10$             ;GO SAVE LOADER
6799  027170  017616  000000  16$:    MOV     @(SP),(SP)      ;FIX UP NO-SAVE RETURN PC
6800  027174  105037  003135          CLRB    XDPSVD          ;CLEAR THE SAVE INDICATOR
6801  027200  000673                  BR      5$              ;BR TO RETURN
6802
6803
6804
6805                                  ;;*****************************************************************
6806                                  ;* SAVXDP - SAVE THE XXDP LOADER IN HI MEMORY
6807                                  ;*THIS SUBROUTINE MOVES THE XXDP LOADER, WHICH RESIDES AT THE
6808                                  ;*PHYSICAL ADDRESS STORED IN XXDPAD, TO THE PHYSICAL ADDRESS
6809                                  ;*STORED IN XDPSAV AND XDPSAV+2. A TOTAL OF 1536(DEC) WORDS ARE MOVED.
6810                                  ;*
6811                                  ;* GETXDP - RESTORE THE XXDP LOADER TO ORIGINAL LOCATION
6812                                  ;*THIS SUBROUTINE MOVES THE RELOCATED XXDP LOADER FROM THE ADDRESS
6813                                  ;*STORED IN XDPSAV,XDPSAV+2, BACK TO LOCATION XXDPAD (ITS ORIGINAL
6814                                  ;*ADDRESS).
6815                                  ;;*****************************************************************
6816  027202  104407  SAVXDP: SAVREG                          ;SAVE R0-R5
6817  027204  005004          CLR     R4                      ;SET INDICATOR TO SAVE XXDP
6818  027206  000410          BR      XDP1
6819  027210  104407  GETXDP: SAVREG                          ;SAVE R0-R5
6820  027212  105037  003134          CLRB    XOVLAD          ;CLEAR XXDP OVERLAID INDICATOR
6821  027216  105737  003135          TSTB    XDPSVD          ;SEE IF XXDP WAS SAVED
6822  027222  001425                  BEQ     XDP2            ;BR IF NOT SAVED
6823  027224  012704  000001          MOV     #1,R4           ;SET INDICATOR TO GET XXDP
6824  027230  105737  000041  XDP1:   TSTB    @#41            ;SEE IF LOADED BY XXDP
6825  027234  001420                  BEQ     XDP2            ;BR IF NOT
6826  027236  012702  003000          MOV     #1536.,R2       ;GET SET TO MOVE 1536(DEC) WORDS
6827  027242  013703  005546          MOV     XXDPAD,R3       ;GET ORIG ADR OF START OF XXDP
6828  027246  013700  005550          MOV     XDPSAV,R0       ;GET SAVE ADR LO BITS
6829  027252  013701  005552          MOV     XDPSAV+2,R1     ;HI BITS
6830  027256  005737  055662          TST     $KT11           ;SEE IF MEM MGT PRESENT
6831  027262  100417                  BMI     XDP3            ;BR IF PRESENT
6832  027264  005704                  TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
6833  027266  001005                  BNE     XDP4            ;BR IF WANT TO GET XXDP
6834  027270  012320  6$:     MOV     (R3)+,(R0)+             ;SAVE A WORD
6835  027272  005302          DEC     R2                      ;SEE IF 1536(DEC) WORDS YET
6836  027274  001375          BNE     6$                      ;BR IF NOT YET
6837  027276  104410  XDP2:   RESREG                          ;RESTORE R0-R5
6838  027300  000207          RTS     PC                      ;RETURN
6839  027302  062700  006000  XDP4:   ADD     #6000,R0        ;POINT TO END OF SAVE AREA
6840  027306  062703  006000          ADD     #6000,R3        ;POINT TO END OF XXDP LOADER AREA
6841  027312  014043  8$:      MOV     -(R0),-(R3)             ;GET A WORD
6842  027314  005302          DEC     R2                      ;SEE IF 1536(DEC) WORDS YET
6843  027316  001375          BNE     8$                      ;BR IF NOT YET
6844  027320  000766          BR      XDP2                    ;GO EXIT
6845                          ;COME HERE IF MEM MGT
6846  027322  004737  026350  XDP3:   JSR     PC,INITMM       ;INIT MEM MGT REGISTERS
6847  027326  006100                  ROL     R0              ;GET ADR BITS 13-21 INTO R1 BITS 7-15
6848  027330  006101                  ROL     R1
6849  027332  006100                  ROL     R0
6850  027334  006101                  ROL     R1
```

```
6851  027336  000301                        SWAB    R1
6852  027340  006100                        ROL     R0
6853  027342  106001                        RORB    R1
6854  027344  010137  172354                MOV     R1,@#KIPAR6     ;SET UP PAR6
6855  027350  013701  005550                MOV     XDPSAV,R1       ;GET LO ADRS BITS
6856  027354  042701  160000                BIC     #160000,R1      ;FIX UP R1 TO REFERENCE PAR6
6857  027360  052701  140000                BIS     #140000,R1
6858  027364  005704              16$:      TST     R4              ;SEE IF WANT TO SAVE OR GET XXDP
6859  027366  001007                        BNE     18$             ;BR IF WANT TO GET XXDP
6860  027370  012305                        MOV     (R3)+,R5        ;SAVE A WORD
6861  027372  005237  177572                INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
6862  027376  010521                        MOV     R5,(R1)+
6863  027400  005337  177572                DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
6864  027404  000406                        BR      20$
6865  027406  005237  177572        18$:    INC     @#SR0           ;TURN ON MEMORY MANAGEMENT
6866  027412  012105                        MOV     (R1)+,R5        ;GET A WORD
6867  027414  005337  177572                DEC     @#SR0           ;TURN OFF MEMORY MANAGEMENT
6868  027420  010523                        MOV     R5,(R3)+
6869  027422  032701  020000        20$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO PAGE 7
6870  027426  001405                        BEQ     22$             ;BR IF NOT
6871  027430  042701  020000                BIC     #BIT13,R1       ;SET PAGE = 5 AS IN
6872  027434  062737  000200  172354        ADD     #200,@#KIPAR6   ;UPDATE PAR BY 4K
6873  027442  005302              22$:      DEC     R2              ;SEE IF 1536 WORDS YET
6874  027444  001347                        BNE     16$             ;BR IF NOT YET
6875  027446  000713                        BR      XDP2            ;BR TO RETURN
6876
6877
6878
6879                                  ;;*****************************************************************
6880                                  .SBTTL  INITSS - INITIALIZE SUBSYSTEM
6881                                  ;*
6882                                  ;*THIS SUBROUTINE INITIALIZES THE DRIVER AND ITS PARAMETERS
6883                                  ;*AND DOES A SUBSYSTEM CLEAR.
6884                                  ;* USES -   R2,R5
6885                                  ;* CALL:
6886                                  ;*              JSR     PC,INITSS
6887                                  ;;*****************************************************************
6888
6889  027450  012737  042130  003046  INITSS: MOV     #ERRHDL,A.ABNL  ;SET UP ABNORMAL ERROR RETURN ADDRESS
6890  027456  012737  040652  003044          MOV     #ERRFRE,A.NORM  ;SET UP NORMAL RETURN ADDRESS
6891  027464  013702  003036                 MOV     RKBAS,R2        ;GET ADDRESS OF RK611 REGISTERS
6892  027470  012705  002630                 MOV     #PARM0,R5       ;GET ADDRESS OF PARAMETER BLOCK
6893  027474  105037  003143                 CLRB    NORTRY          ;CLEAR "NO-RETRY" FLAG
6894  027500  004737  027556                 JSR     PC,CLRPRM       ;CLEAR DRIVER INPUT PARAMETERS
6895  027504  112765  000177  00.00'         MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
6896  027512  004737  040252                 JSR     PC,DRVCAL       ;DO SUBSYSTEM CLEAR
6897  027516  113765  005510  000000         MOVB    DRIVE,P.DRVN(R5) ;SET CURRENT DRIVE NO.
6898  027524  113737  005510  002714         MOVB    DRIVE,PARM1     ;SET DRIVE NO. IN ALTERNATE P.B.
6899  027532  113765  003125  000007         MOVB    FORMAT,P.CS1H(R5) ;SET CURRENT DRIVE FORMAT
6900  027540  153765  003124  000007         BISB    TYPFMT,P.CS1H(R5)       ;SET DRV TYPE BIT
6901  027546  116537  000007  002723         MOVB    P.CS1H(R5),PARM1+P.CS1H   ;COPY DR. TYPE & FORMAT INTO PARM1
6902  027554  000207                         RTS     PC              ;SUBROUTINE EXIT
6903
6904
6905
6906                                  ;;*****************************************************************
```

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

027556  010046                  CLRPRM: MOV     R0,-(SP)        ;SAVE R0
027560  010546                          MOV     R5,-(SP)        ;SAVE R5
027562  010500                          MOV     R5,R0           ;GET PARAMETER BLOCK ADDRESS
027564  062705  000016                  ADD     #P.CS1,R5       ;COMPUTE LIMIT ADDRESS
027570  005020                  1$:     CLR     (R0)+           ;CLEAR A WORD IN PARAMETER BLOCK
027572  020005                          CMP     R0,R5           ;SEE IF DONE YET
027574  001375                          BNE     1$              ;BR IF  NOT DONE  YET
027576  012605                          MOV     (SP)+,R5        ;RESTORE R5
027600  012600                          MOV     (SP)+,R0        ;RESTORE R0
027602  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>
                                        ;*
                                        ;;*****************************************************************
027604  105037  003124          SCNDRV: CLRB    TYPFMT          ;ASSUME RK06 FOR FIRST TRY
027610  004737  027450                  JSR     PC,INITSS       ;INITIALIZE DRIVER AND CLEAR SUBSYSTEM
027614  113765  005510  000000        MOVB    DRIVE,P.DRVN(R5) ;GET DRIVE NUMBER
                                ;SEE IF DRIVE 0 IS XXDP LOAD MEDIUM
027622  001012                          BNE     3$              ;BR IF NOT DRIVE 0
027624  122737  000013  000041        CMPB    #13,@#41        ;SEE IF RK06 IS XXDP MEDIUM
027632  001006                          BNE     3$              ;BR IF NOT
027634  105737  003116                TSTB    MDFLAG          ;SEE IF 200 START
027640  001003                          BNE     3$              ;BR IF NOT
027642  104401  010075                TYPE    ,DR0XDP         ;TYPE "DRIVE 0 IS LOAD MEDIUM"
027646  000517                          BR      2$              ;TAKE ERROR EXIT
027650  042712  000100          3$:     BIC     #IE,(R2)        ;DISABLE RK06 INTERRUPT
027654  113762  005510  000010        MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
027662  105737  003124                TSTB    TYPFMT          ;SEE IF RK07
027666  001003                          BNE     5$              ;BR IF YES
027670  012712  000001                MOV     #GO,(R2)        ;ELSE SET GO FOR RK06 IN CS1
027674  000402                          BR      7$
027676  012712  002001          5$:     MOV     #<CDT!GO>,(R2)  ;SET GO FOR RK07 IN CS1
027702  005037  005542          7$:     CLR     SCRACH          ;CLEAR STALL CTR
027706  005237  005542          14$:    INC     SCRACH          ;INCREMENT STALL COUNTER
027712  001375                          BNE     14$             ;STALL FOR SEVERAL MILLI-SEC
```

```
 6963  027714  032762  001000  000010          BIT     #MDS,RKCS2(R2)  ;SEE IF MDS ERROR
 6964  027722  001417                          BEQ     18$             ;BR IF NOT
 6965  027724  112765  000101  000001          MOVB    #SELDRV,P.CMND(R5) ;SET CMND FOR REPORT
 6966  027732  011265  000016                  MOV     (R2),P.CS1(R5)  ;GET RKCS1
 6967  027736  004737  050156                  JSR     PC,I.CST1       ;GET OTHER RK611 REGISTERS
 6968  027742  004737  041650                  JSR     PC,REPSUP       ;SET UP ERROR REPORT
 6969  027746  104052                          ERROR   52              ;REPORT MDS ERROR
 6970  027750  052737  000200  005504          BIS     #ABORT,RECODE   ;SET ABORT FLAG
 6971  027756  000137  043724                  JMP     ALLTRM          ;GO ABORT TESTING
 6972  027762  032762  000040  000014  18$:     BIT     #DTYE,RKER(R2)  ;SEE IF DRV TYPE ERROR
 6973  027770  001403                          BEQ     20$             ;BR IF NO, =RK06
 6974  027772  152737  000004  003124          BISB    #B.CDT,TYPFMT   ;ELSE SET RK07 FLAG
 6975  030000  004737  027450          20$:    JSR     PC,INITSS       ;INIT THE SUBSYSTEM
 6976  030004  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ DRIVE STATUS COMMAND
 6977  030012  012737  025550  003046          MOV     #NEDHDL,A.ABNL  ;SET NED ABNORMAL RETURN ADDRESS
 6978  030020  012737  000377  005540          MOV     #377,NEWON      ;INIT. ON-LINE INDICATOR
 6979  030026  004737  040252                  JSR     PC,DRVCAL       ;READ ALL DRIVE STATUS
 6980                          ;SEE IF THIS DRIVE EXISTS AND IS ON-LINE
 6981  030032  012737  042130  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ABNL RETURN ADDRESS
 6982  030040  022737  000377  005540          CMP     #377,NEWON      ;SEE IF NED INDICATION ON THIS DRIVE
 6983  030046  001425                          BEQ     4$              ;BR IF NED NOT SET
 6984  030050  113737  005510  007617          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
 6985  030056  152737  000060  007617          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
 6986  030064  104401  007611                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
 6987  030070  104401  007622                  TYPE    ,NXDRIV         ;TYPE "NON-EXISTENT"
 6988  030074  132737  000200  001341          BITB    #BIT7,$ENVM     ;SEE IF UNDER APT
 6989  030102  001401                          BEQ     2$              ;BR IF NO
 6990  030104  104123                          ERROR   123             ;NED UNDER APT SIZING
 6991                          ;SERVICE ERRORS HERE
 6992  030106  042762  000100  000000  2$:     BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
 6993  030114  017616  000000                  MOV     @(SP),(SP)      ;SET UP ERROR RETURN ADDRESS
 6994  030120  000207                          RTS     PC              ;ERROR RETURN
 6995                          ;SEE IF DRIVE IS READY
 6996  030122  032765  000200  000040  4$:     BIT     #S.DRY,P.A00(R5) ;TEST FOR DRIVE READY
 6997  030130  001013                          BNE     6$              ;BR IF DRIVE IS READY
 6998  030132  113737  005510  007617          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO. INTO BUF
 6999  030140  152737  000060  007617          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
 7000  030146  104401  007611                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
 7001  030152  104401  007636                  TYPE    ,NTREDY         ;TYPE "NOT READY"
 7002  030156  000753                          BR      2$              ;TAKE ERROR EXIT
 7003                          ;SEE IF DRIVE IS WRITE ENABLED
 7004  030160  032765  004000  000040  6$:     BIT     #S.WRL,P.A00(R5) ;SEE IF WRITE LOCK SET
 7005  030166  001413                          BEQ     8$              ;BR IF WRITE LOCK NOT SET
 7006  030170  113737  005510  007617          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
 7007  030176  152737  000060  007617          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
 7008  030204  104401  007611                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
 7009  030210  104401  007650                  TYPE    ,WRTLOK         ;TYPE "WRITE-LOCKED"
 7010  030214  000734                          BR      2$              ;TAKE ERROR EXIT
 7011                          ;SEE IF DRIVE NOT LOADED WITH ALIGNMENT CARTRIDGE
 7012  030216  112765  000103  000001  8$:     MOVB    #PACK,P.CMND(R5) ;SET PACK ACKNOWLEDGE COMMAND
 7013  030224  004737  040252                  JSR     PC,DRVCAL       ;SET VOLUME VALID
 7014  030230  112765  000113  000001          MOVB    #RECAL,P.CMND(R5)       ;SET RECAL COMMAND
 7015  030236  004737  040252                  JSR     PC,DRVCAL       ;RECALIBRATE DRIVE
 7016  030242  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 7017  030250  105737  003124                  TSTB    TYPFMT          ;SEE IF RK07
 7018  030254  001004                          BNE     25$             ;BR IF YES
```

```
7019  030256  012765  000632  000002          MOV     #632,P.CYLN(R5)
7020  030264  000403                          BR      27$
7021  030266  012765  001456  000002  25$:    MOV     #1456,P.CYLN(R5)
7022  030274  112765  000002  000005  27$:    MOVB    #LSTTRK,P.TRCK(R5)      ;SET TRACK=2
7023  030302  012765  063520  000010          MOV     #RWBUF,P.BALO(R5) ;BUS ADDRESS
7024  030310  012765  177774  000012          MOV     #-4,P.WC(R5)    ;READ 4 WORDS
7025  030316  142765  000020  000007          BICB    #8.CFMT,P.GS1H(R5)      ;SET 22 SECTOR FORMAT
7026  030324  153765  003124  000007          BISB    TYPFMT,P.CS1H(R5)       ;SET CORRECT DRV TYPE
7027  030332  004737  040252                  JSR     PC,DRVCAL       ;READ 4 WORDS OF BAD SECTOR FILE
7028  030336  032737  100000  005504          BIT     #ANYDER,RECODE  ;SEE IF DATA ERROR
7029  030344  001402                          BEQ     10$             ;BR IF OK
7030  030346  104401  012307                  TYPE    ,BAD632         ;TYPE READ ERROR MESSAGE
7031  030352  022737  177777  063526  10$:    CMP     #177777,RWBUF+6 ;SEE IF ALL 1'S IN I.D. WORD 3
7032  030360  001013                          BNE     12$             ;BR IF NOT ALL 1'S
7033  030362  113737  005510  007617          MOVB    DRIVE,BADDRV+6  ;GET DRIVE NO.
7034  030370  152737  000060  007617          BISB    #'0,BADDRV+6    ;CONVERT TO ASCII
7035  030376  104401  007611                  TYPE    ,BADDRV         ;TYPE "DRIVE X"
7036  030402  104401  007663                  TYPE    ,ALNPAK         ;TYPE "LOADED WITH ALIGN PACK"
7037  030406  000637                          BR      2$              ;TAKE ERROR EXIT
7038                                  ;ERROR FREE RETURN
7039  030410  062716  000002          12$:    ADD     #2,(SP)         ;FIX UP RETURN PC
7040  030414  000207                          RTS     PC              ;RETURN
7041
7042
7043
7044                                  ;;*****************************************************************
7045                                  ;*SETTUP - SET UP FOR LOOP ON ERROR
7046                                  ;*THIS SUBROUTINE CANNOT BE CALLED BY ANY OTHER
7047                                  ;*SUBROUTINE --- ONLY MAIN-LINE CODE !!!!
7048                                  ;;*****************************************************************
7049  030416  011637  001076          SETTUP: MOV     (SP),@#STACK-2  ;MOVE RETURN PC ON STACK
7050  030422  012706  001076                  MOV     #STACK-2,SP     ;RE-INIT THE STACK POINTER
7051  030426  005037  005504                  CLR     RECODE          ;CLEAR ERROR RECOVERY FLAGS
7052  030432  105037  003126                  CLRB    ERRCNT          ;CLEAR RETRY ERROR COUNT
7053  030436  012705  002630                  MOV     #PARM0,R5       ;SET PARAM BLK ADRS
7054  030442  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR CMND
7055  030450  004737  040252                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
7056  030454  012737  000000  177776          MOV     #PR0,@#PS       ;RE-ESTABLISH PRIORITY 0
7057  030462  000207                          RTS     PC              ;RETURN
7058
7059
7060
7061                                  ;;*****************************************************************
7062                                  .SBTTL  CHKITR - CHECK CURRENT TEST ITERATION NUMBER
7063                                  ;*THIS SUBROUTINE LOADS THE ITERATION NUMBER FOR THE CURRENT
7064                                  ;*TEST INTO $TIMES, AND CHECKS IT TO DETERMINE IF THIS TEST
7065                                  ;*SHOULD BE RUN. IF ITERATION NO. = 0, THE SUBROUTINE RETURNS
7066                                  ;*TO THE STORED PC (AND THE TEST WILL BE SKIPPED). IF IT I
7067                                  ;*NOT 0, THE RETURN IS MADE TO PC+2, AND THE TEST WILL
7068                                  ;*BE RUN. HOWEVER, ON THE FIRST PASS (QUICK VERIFY) OF A RUN
7069                                  ;*WHICH STARTED AT ADRS 200, ALL TESTS WILL BE RUN ONCE.
7070
7071  030464  010146          CHKITR: MOV     R1,-(SP)        ;SAVE R1
7072  030466  112737  000144  001115          MOVB    #100.,$ERMAX    ;SET MAX ERROR CNT TO 100 FOR $SCOPE
7073  030474  105737  003136                  TSTB    DULACS          ;SEE IF DUAL-ACCESS FLAG SET
7074  030500  001006                          BNE     3$              ;BR IF YES
```

```
7075  030502  105737  003116            TSTB    MDFLAG          ;SEE IF 200 START
7076  030506  001007                    BNE     4$              ;BR IF NOT
7077  030510  005737  001326            TST     $PASS           ;SEE IF FIRST PASS
7078  030514  001004                    BNE     4$              ;BR IF NOT
7079  030516  012737  000001  001304 3$:  MOV     #1,$TIMES       ;SET UP FOR 1 ITERATION
7080  030524  000410                    BR      1$              ;GO RUN DUAL-ACCESS TEST
7081  030526  113701  001102        4$:  MOVB    $TSTNM,R1       ;GET CURRENT TEST NUMBER
7082  030532  005301                    DEC     R1              ;DECREMENT BY 1
7083  030534  006301                    ASL     R1              ;DOUBLE IT, TO GET TEST LIST INDEX
7084  030536  016137  005626  001304    MOV     TSTLST(R1),$TIMES ;LOAD ITERATION NUMBER
7085  030544  001403                    BEQ     2$              ;BR IF TEST SHOULD BE SKIPPED
7086  030546  062766  000004  000002 1$:  ADD     #4,2(SP)        ;ADJUST RETURN PC TO RUN THIS TEST
7087  030554  012601                2$:  MOV     (SP)+,R1        ;RESTORE R1
7088  030556  000207                    RTS     PC              ;RETURN
7089
7090
7091
7092                                ;;*****************************************************************
7093                                .SBTTL  RDCHRS - READ A STRING OF KBD INPUT CHARS
7094                                ;*THIS SUBROUTINE READS A STRING OF UP TO EIGHTY INPUT
7095                                ;*CHARACTERS AT THE KBD, TERMINATED BY <CR>, AND PLACES
7096                                ;*THEM IN BUFFO, TERMINATED BY A NULL (0) BYTE.
7097                                ;*RUB-OUT AND (^U) FEATURES ARE PROVIDED.
7098                                ;*IMMEDIATELY FOLLOWING THE SUBROUTINE CALL, THREE SPECIAL
7099                                ;*RETURN ADDRESSES MUST BE LISTED: THE FIRST RETURN IS
7100                                ;*TAKEN IF (^C) IS TYPED, THE SECOND IS TAKEN IF (^Z) IS
7101                                ;*TYPED, AND THE THIRD IS TAKEN IF (^U) OR INVALID INPUT IS TYPED.
7102                                ;*IF (^G) IS TYPED, THE SOFTWARE SWITCH REGISTER IS OPENED
7103                                ;*FOR MODIFICATION, IF SELECTED, AND THEN THE (^G) RETURN
7104                                ;*IS TAKEN.
7105                                ;*THE NUMBER OF INPUT CHARACTERS IN THE BUFFER IS RETURNED
7106                                ;*IN R0.
7107                                ;*
7108                                ;*  CALL -      JSR     PC,RDCHRS
7109                                ;*              <CONTROL-C RETURN ADDRESS>
7110                                ;*              <CONTROL-Z RETURN ADDRESS>
7111                                ;*              <CONTROL-U OR ERROR RETURN ADDRESS>
7112                                ;*              RETURN
7113                                ;*****************************************************************
7114
7115  030560                        RDCHRS:
7116  030560  010146                        MOV     R1,-(SP)        ;SAVE R1
7117  030562  010246                        MOV     R2,-(SP)        ;SAVE R2
7118  030564  005000                        CLR     R0              ;INITIALIZE CHARACTER COUNT
7119  030566  005001                        CLR     R1              ;INITIALIZE RUB-OUT INDICATOR
7120                                ;READ A CHARACTER
7121  030570  104406                2$:     RDCHR                   ;READ A CHARACTER
7122  030572  112602                        MOVB    (SP)+,R2        ;GET CHARACTER INTO R2
7123                                ;CHECK FOR (^C)
7124  030574  122702  000003               CMPB    #003,R2         ;SEE IF (^C) TYPED
7125  030600  001006                        BNE     4$              ;BR IF NOT (^C)
7126  030602  104401  012353               TYPE    ,CNTRLC         ;ECHO (^C)
7127  030606  017666  000004  000004 3$:     MOV     @4(SP),4(SP)    ;PUT RETURN ADDRESS ON STACK
7128  030614  000523                        BR      24$             ;BR TO TAKE EXIT
7129                                ;CHECK FOR (^Z)
7130  030616  122702  000032        4$:     CMPB    #032,R2         ;SEE IF (^Z) TYPED
```

```
7131  030622  001006                          BNE     6$              ;BR IF NOT (^Z)
7132  030624  104401  012360                  TYPE    ,CNTRLZ         ;ECHO (^Z)
7133  030630  062766  000002  000004          ADD     #2,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADR.
7134  030636  000763                          BR      3$              ;BR TO TAKE (^Z) EXIT
7135                                  ;CHECK FOR (^U)
7136  030640  122702  000025          6$:     CMPB    #025,R2         ;SEE IF (^U) TYPED
7137  030644  001006                          BNE     8$              ;BR IF NOT (^U)
7138  030646  104401  012372                  TYPE    ,CNTRLU         ;ECHO (^U)
7139  030652  062766  000004  000004  7$:     ADD     #4,4(SP)        ;MAKE OLD PC POINT TO NEXT RETURN ADDR.
7140  030660  000752                          BR      3$              ;BR TO TAKE (^U) EXIT
7141                                  ;CHECK FOR (^G)
7142  030662  122702  000007          8$:     CMPB    #007,R2         ;SEE IF (^G) TYPED
7143  030666  001005                          BNE     9$              ;BR IF NOT (^G)
7144  030670  104401  012377                  TYPE    ,CNTRLG         ;ECHO (^G)
7145  030674  004737  026316                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWITCH REG. FOR CHANGE
7146  030700  000764                          BR      7$              ;TAKE (^U) RETURN
7147                                  ;CHECK FOR RUB-OUT (DELETE)
7148  030702  122702  000177          9$:     CMPB    #177,R2         ;SEE IF RUB-OUT (DEL) TYPED
7149  030706  001020                          BNE     14$             ;BR IF NOT RUB-OUT
7150  030710  005700                          TST     R0              ;CHECK THE CHARACTER COUNT
7151  030712  001726                          BEQ     2$              ;BR IF COUNT = 0
7152  030714  005701                          TST     R1              ;CHECK THE RUB-OUT INDICATOR
7153  030716  001003                          BNE     11$             ;BR IF WE HAD A PREVIOUS RUB-OUT
7154  030720  005201                          INC     R1              ;SET RUB-OUT INDICATOR
7155  030722  104401  012410                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH (\)
7156  030726  005037  005542          11$:    CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
7157  030732  005300                          DEC     R0              ;DECREMENT COUNT
7158  030734  116037  005272  005542          MOVB    BUFFO(R0),SCRACH ;GET LAST CHAR. INTO BUFFER
7159  030742  104401  005542                  TYPE    ,SCRACH         ;ECHO CHARACTER TO BE DELETED
7160  030746  000710                          BR      2$              ;GO READ ANOTHER CHARACTER
7161  030750  005701                  14$:    TST     R1              ;CHECK THE RUB-OUT INDICATOR
7162  030752  001403                          BEQ     16$             ;BR IF INDICATOR IS NOT SET
7163  030754  104401  012410                  TYPE    ,BKSLSH         ;TYPE A BACK-SLASH
7164  030760  005001                          CLR     R1              ;CLEAR THE RUB-OUT INDICATOR
7165                                  ;CHECK FOR CARRIAGE RETURN
7166  030762  122702  000015          16$:    CMPB    #015,R2         ;SEE IF <CR> TYPED
7167  030766  001426                          BEQ     19$             ;BR IF <CR>
7168                                  ;HANDLE POSSIBLE DIGIT
7169  030770  005037  005542                  CLR     SCRACH          ;USE SCRATCH WORD FOR TEMP. BUFFER
7170  030774  110237  005542                  MOVB    R2,SCRACH       ;GET THIS CHARACTER INTO BUFFER
7171  031000  104401  005542                  TYPE    ,SCRACH         ;ECHO THE CHARACTER TYPED
7172  031004  110260  005272                  MOVB    R2,BUFFO(R0)    ;PUT CHARACTER INTO BUFFER
7173  031010  005200                          INC     R0              ;INCREMENT CHARACTER COUNTER
7174  031012  022700  000120                  CMP     #80.,R0         ;SEE IF TOO MANY CHARACTERS TYPED
7175  031016  001264                          BNE     2$              ;BR IF NOT TOO MANY
7176  031020  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
7177  031024  112760  000000  005272          MOVB    #0,BUFFO(R0)  ;PUT TERMINATING NULL INTO BUFFER
7178  031032  104401  005272                  TYPE    ,BUFFO          ;ECHO INPUT STRING
7179  031036  104401  001314                  TYPE    ,$QUES          ;TYPE <?>,<CR>,<LF>
7180  031042  000703                          BR      7$              ;TAKE ERROR EXIT FROM RDCHRS
7181  031044  104401  001315          19$:    TYPE    ,$CRLF          ;TYPE <CR>,<LF>
7182  031050  112760  000000  005272          MOVB    #0,BUFFO(R0)    ;PUT TERMINATING NULL INTO BUFFER
7183  031056  062766  000006  000004          ADD     #6,4(SP)        ;FIX UP RETURN ADDRESS
7184  031064  012602                  24$:    MOV     (SP)+,R2        ;RESTORE R2
7185  031066  012601                          MOV     (SP)+,R1        ;RESTORE R1
7186  031070  000207                          RTS     PC              ;SUBROUTINE EXIT
```

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

031072  104401  012421          TYPTST: TYPE    ,SPACE1          ;TYPE A SPACE
031076  010146                          MOV     R1,-(SP)         ;PUT INDEX ONTO STACK
031100  006216                          ASR     (SP)             ;DIVIDE BY 2
031102  005216                          INC     (SP)             ;INCREMENT TO GET TEST NO.
031104  104403                          TYPOS                    ;TYPE TEST NO. IN OCTAL
031106     002                          .BYTE   2                ;TYPE 2 DIGITS
031107     000                          .BYTE   0                ;SUPPRESS LEADING ZEROS
031110  104401  012414                  TYPE    ,SPACE6          ;TYPE 6 SPACES
031114  016146  005626                  MOV     TSTLST(R1),-(SP) ;PUT CURRENT ITERATION NO. ONTO STACK
031120  104403                          TYPOS                    ;TYPE ITERATION NO. IN OCTAL
031122     006                          .BYTE   6                ;TYPE 6 DIGITS
031123     000                          .BYTE   0                ;SUPPRESS LEADING ZEROS
031124  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
                                        ;;*****************************************************************

031126  016137  006152  012424  TYPPRM: MOV     PRMNEM(R1),PRMBUF ;PUT MNEMONIC INTO OUTPUT BUFFER
031134  104401  012424                  TYPE    ,PRMBUF          ;TYPE "XX="
031140  016137  005732  005476          MOV     PRMLST(R1),LOWOCT ;PUT LOW BITS OF PARAM. INTO LOWOCT
031146  005037  005500                  CLR     HIGOCT           ;CLEAR HIGH BINARY BITS
031152  022761  040515  006152          CMP     #'MA,PRMNEM(R1)  ;SEE IF PARAMETER IS (MA)
031160  001003                          BNE     1$               ;BR IF NOT (MA)
031162  016137  005734  005500          MOV     PPMLST+2(R1),HIGOCT ;PUT HI BITS OF PARAM. INTO HIGOCT
031170  012746  005476          1$:     MOV     #LOWOCT,-(SP)    ;PUT ADDRESS OF BINARY VALUE ON STACK
031174  004737  053366                  JSR     PC,@#$DB2O       ;CONVERT BINARY TO OCTAL ASCII
031200  004737  053702                  JSR     PC,@#$SUPRS      ;TYPE YYYYYYYY, SUPPRESSING LEADING 0'S
031204  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
                        ;;*****************************************************************

031206                  TYPPAT:
031206  104401  012430          TYPE    ,WORDSP         ;TYPE 'WORD ''
031212  010146                  MOV     R1,-(SP)        ;PUT INDEX ONTO STACK
031214  006216                  ASR     (SP)            ;DIVIDE BY TWO FOR WORD NO.
031216  104403                  TYPOS                   ;GO TYPE WORD NUMBER
031220     002                  .BYTE   2               ;DIGIT COUNT = 2 FOR TYPOC
031221     001                  .BYTE   1               ;TELL TYPOS TO TYPE LEADING ZEROS
031222  104401  012436          TYPE    ,EQUALS         ;TYPE '' = ''
031226  016146  007156          MOV     PAT15(R1),-(SP) ;GET BINARY VALUE OF THIS WORD
031232  104403                  TYPOS                   ;TYPE VALUE IN OCTAL
031234     006                  .BYTE   6               ;TYPE 6 DIGITS
031235     001                  .BYTE   1               ;TYPE LEADING ZEROS
031236  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
                        ;;*****************************************************************

031240                  MODP15:
031240  010046                  MOV     R0,-(SP)        ;SAVE R0
031242  010146                  MOV     R1,-(SP)        ;SAVE R1
031244  010246                  MOV     R2,-(SP)        ;SAVE R2
                        ;SEE IF PARAMETER IS PT
031246  022761  052120  006152          CMP     #''PT,PRMNEM(R1) ;SEE IF CURRENT PARAMETER IS (PT)
031254  001127                  BNE     22$             ;BR IF NOT (PT)
                        ;SEE IF PATTERN 15 IS SPECIFIED
031256  032737  100000  005760          BIT     #BIT15,PT       ;SEE IF PATTERN 15 SPECIFIED
031264  001523                  BEQ     22$             ;BR IF NOT SPECIFIED
                        ;SEE IF PATTERN 15 SHOULD BE MODIFIED
031266  104401  011130  4$:     TYPE    ,MDFY15         ;ASK WHETHER PATTERN 15 SHOULD BE MODIFIED
031272  004737  030560          JSR     PC,RDCHRS       ;READ RESPONSE
031276  031544                  24$                     ;(^C) RETURN ADDRESS
031300  031554                  26$                     ;(^Z) RETURN ADDRESS
031302  031340                  8$                      ;(^U) OR ERROR RETURN ADDRESS
031304  005700                  TST     R0              ;SEE IF NULL INPUT
031306  001512                  BEQ     22$             ;BR IF MODIFICATION NOT REQUESTED
```

```
 7299  031310  022737  000115  005272          CMP     #'M,BUFFO       ;SEE IF (M) TYPED
 7300  031316  001405                          BEQ     6$              ;BR IF MODIFICATION REQUESTED
 7301  031320  104401  005272                  TYPE    ,BUFFO          ;ECHO BAD INPUT
 7302  031324  104401  001314                  TYPE    ,$QUES
 7303  031330  000756                          BR      4$              ;GO ASK AGAIN
 7304                              ;MODIFY PATTERN 15
 7305  031332  104401  011106          6$:     TYPE    ,SELP15         ;TYPE 'MODIFY PATTERN 15''
 7306  031336  005001                          CLR     R1              ;INITIALIZE WORD INDEX
 7307  031340  004737  031206          8$:     JSR     PC,TYPPAT       ;TYPE CURRENT WORD AND VALUE
 7308  031344  104401  012421                  TYPE    ,SPACE1         ;TYPE A SPACE
 7309  031350  104401  012442                  TYPE    ,PROMPT         ;TYPE ASTERISK AND SPACE
 7310                              ;READ AND CHECK INPUT, IF ANY
 7311  031354  004737  030560                  JSR     PC,RDCHRS       ;READ NEW DATA PATTERN WORD
 7312  031360  031544                          24$                     ;(^C) RETURN ADDRESS FOR RDCHRS
 7313  031362  031554                          26$                     ;(^Z) RETURN ADDRESS FOR RDCHRS
 7314  031364  031340                          8$                      ;(^U) OR ERROR RETURN ADDR. FOR RDCHRS
 7315  031366  005700                          TST     R0              ;SEE IF ANY INPUT
 7316  031370  001006                          BNE     12$             ;BR IF ANY INPUT
 7317  031372  062701  000002          10$:    ADD     #2,R1           ;INCREMENT WORD INDEX
 7318  031376  022701  000040                  CMP     #32.,R1         ;SEE IF ALL DONE
 7319  031402  001454                          BEQ     22$             ;BR IF DONE, TO RETURN
 7320  031404  000755                          BR      8$              ;CONTINUE WITH NEXT WORD
 7321  031406  022737  000041  005272  12$:    CMP     #'.,BUFFO       ;SEE IF (!) TYPED
 7322  031414  001431                          BEQ     16$             ;BR TO PROPAGATE CURRENT VALUE
 7323  031416  005002                          CLR     R2              ;INIT. (!) INDICATOR
 7324  031420  122760  000041  005271          CMPB    #'!,BUFFO-1(R0) ;SEE IF LAST CHAR IN BUF IS (.)
 7325  031426  001004                          BNE     14$             ;BR IF NOT (!)
 7326  031430  105060  005271                  CLRB    BUFFO-1(R0)     ;INSERT TERMINATOR BYTE
 7327  031434  005300                          DEC     R0              ;DECREMENT CHAR COUNT
 7328  031436  005202                          INC     R2              ;SET (!) INDICATOR
 7329  031440  022700  000006          14$:    CMP     #6,R0           ;SEE HOW MANY CHARS NOW
 7330  031444  002426                          BLT     20$             ;BR IF TOO MANY
 7331  031446  012746  005272                  MOV     #BUFFO,-(SP)    ;GET BUF. ADDR. ON STACK FOR OCTBIN
 7332  031452  004737  052430                  JSR     PC,OCTBIN       ;CHECK DIGITS AND CONVERT TO BINARY
 7333  031456  031522                          20$                     ;ERROR RETURN ADDRESS FOR OCTBIN
 7334  031460  012600                          MOV     (SP)+,R0        ;GET BINARY VALUE
 7335  031462  005737  052562                  TST     $HIOCT          ;SEE IF VALUE EXCEEDS 16 BITS
 7336  031466  001015                          BNE     20$             ;BR TO ECHO BAD INPUT
 7337  031470  010061  007156                  MOV     R0,PAT15(R1)    ;PUT NEW WORD VALUE INTO TABLE
 7338  031474  005702                          TST     R2              ;SEE IF (!) WAS TYPED
 7339  031476  001735                          BEQ     10$             ;BR IF (!) WAS NOT TYPED
 7340                              ;PROPAGATE CURRENT WORD TO END OF PATTERN 15 TABLE
 7341  031500  022701  000036          16$:    CMP     #30.,R1         ;SEE IF ALL DONE YET
 7342  031504  001413                          BEQ     22$             ;BR IF DONE
 7343  031506  016161  007156  007160          MOV     PAT15(R1),PAT15+2(R1) ;PROPAGATE TO NEXT WORD
 7344  031514  062701  000002                  ADD     #2,R1           ;INCREMENT WORD INDEX
 7345  031520  000767                          BR      16$             ;LOOP UNTIL DONE
 7346                              ;ECHO BAD INPUT
 7347  031522  104401  005272          20$:    TYPE    ,BUFFO          ;ECHO BAD INPUT
 7348  031526  104401  001314                  TYPE    ,$QUES          ;TYPE <?> AND <CR>,<LF>
 7349  031532  000702                          BR      8$              ;BR TO ASK AGAIN
 7350                              ;NORMAL RETURN
 7351  031534  012602          22$:    MOV     (SP)+,R2        ;RESTORE R2
 7352  031536  012601                  MOV     (SP)+,R1        ;RESTORE R1
 7353  031540  012600                  MOV     (SP)+,R0        ;RESTORE R0
 7354  031542  000207                  RTS     PC              ;RETURN
```

```
                                        ;(^C) RETURN
031544  012766  013660  000006  24$:    MOV     #DRVTST,6(SP)   ;PC FOR (^C) RETURN
031552  000770                          BR      22$             ;BR TO RETURN
                                        ;(^Z) RETURN
031554  012766  015272  000006  26$:    MOV     #ASKMDE,6(SP)   ;PC FOR (^Z) RETURN
031562  000764                          BR      22$             ;BR TO RETURN


                                        ;;******************************************************************
                                        .SBTTL  CHKPRM - CHECK VALUE OF INPUT PARAMETER
                                        ;*THIS SUBROUTINE CHECKS THE CONTENTS OF LOWOCT AND
                                        ;*HIGOCT (IF 32-BIT VALUE) AGAINST LOWER AND UPPER
                                        ;*LIMITS FOR THE CURRENT PARAMETER. THE PARAMETER TABLE
                                        ;*INDEX FOR THE CURRENT PARAMETER MUST BE PASSED
                                        ;*TO CHKPRM IN R1 ON ENTRY. IF THE PARAMETER VALUE
                                        ;*IS LEGAL, IT IS ENTERED INTO THE PARAMETER TABLE
                                        ;*(PRMLST). IF NOT, A RETURN IS MADE TO THE ADDR. FOLLOWING
                                        ;*THE CALL TO CHKPRM. IF THE PARAMETER HAS A 2-WORD
                                        ;*VALUE (32 BITS) R1 IS INCREMENTED BY 2 BEFORE RETURN.
                                        ;* CALL -      JSR     PC,CHKPRM
                                        ;*            <ERROR RETURN ADDRESS>
                                        ;*            RETURN
                                        ;;******************************************************************

031564  104407                  CHKPRM: SAVREG                  ;SAVE R0-R5
031566  010102                          MOV     R1,R2           ;GET COPY OF INDEX
031570  006302                          ASL     R2              ;DOUBLE IT
031572  022761  040515  006152          CMP     #''MA,PRMNEM(R1) ;SEE IF PARAMETER IS (MA)
031600  001422                          BEQ     20$             ;BR IF (MA)
031602  005737  005500                  TST     HIGOCT          ;SEE IF HIGH BITS = 0
031606  001403                          BEQ     18$             ;BR IF ZERO
031610  017616  000000          16$:    MOV     @(SP),(SP)      ;GET ERROR RETURN PC
031614  000475                          BR      30$             ;GO TO RETURN
                                        ;CHECK VALIDITY OF 16-BIT PARAMETER VALUE
031616  023762  005476  006042  18$:    CMP     LOWOCT,PRMLIM(R2) ;SEE IF INPUT VALUE IS TOO SMALL
031624  103771                          BLO     16$             ;BR IF INPUT VALUE TOO SMALL
031626  023762  005476  006044          CMP     LOWOCT,PRMLIM+2(R2) ;SEE IF INPUT VALUE IS TOO LARGE
031634  101365                          BHI     16$             ;BR IF INPUT VALUE IS TOO LARGE
                                        ;UPDATE 16-BIT PARAMETER VALUE IN LIST
031636  013761  005476  005732          MOV     LOWOCT,PRMLST(R1) ;PUT NEW PARAMETER VALUE INTO LIST
031644  000457                          BR      28$             ;BR TO RETURN
                                        ;CHECK VALIDITY OF 32-BIT PARAMETER VALUE
031646  032737  000001  005476  20$:    BIT     #BIT0,LOWOCT    ;SEE IF MA IS ODD
031654  001355                          BNE     16$             ;BR IF MA IS ODD
031656  023762  005500  006044          CMP     HIGOCT,PRMLIM+2(R2) ;SEE IF HIGH WORD TOO SMALL
031664  103751                          BLO     16$             ;BR IF HIGH WORD IS TOO SMALL
031666  001004                          BNE     24$             ;BR IF HIGH WORD NOT EQUAL TO LOW LIMIT
031670  023762  005476  006042          CMP     LOWOCT,PRMLIM(R2) ;SEE IF LOW WORD IS TOO SMALL
031676  103744                          BLO     16$             ;BR IF LOW WORD IS TOO SMALL
031700  023762  005500  006050  24$:    CMP     HIGOCT,PRMLIM+6(R2) ;SEE IF HIGH WORD IS TOO BIG
031706  101340                          BHI     16$             ;BR IF HIGH WORD TOO BIG
031710  001004                          BNE     26$             ;BR IF HI WORD NOT EQUAL TO UPPER LIMIT
031712  023762  005476  006046          CMP     LOWOCT,PRMLIM+4(R2) ;SEE IF LOW WORD IS TOO LARGE
031720  101333                          BHI     16$             ;BR IF LOW WORD IS TOO LARGE
                                        ;UPDATE 32-BIT PARAMETER VALUE IN LIST
```

```
 7411  031722  013761  005476  005732  26$:    MOV     LOWOCT,PRMLST(R1) ;PUT LOW WORD INTO LIST
 7412  031730  013761  005500  005734          MOV     HIGOCT,PRMLST+2(R1) ;PUT HIGH WORD INTO LIST
 7413                                  ;COMPUTE AND TYPE MAX ALLOWABLE WORD COUNT FOR THIS MA
 7414  031736  004737  032014                  JSR     PC,MXWRDC       ;GET WORD COUNT IN R1-R0
 7415  031742  104401  010651                  TYPE    ,MAWRDC         ;TYPE 'MAX WORD COUNT = ''
 7416  031746  010037  003176                  MOV     R0,SUMLO1
 7417  031752  010137  003200                  MOV     R1,SUMHI1
 7418  031756  012746  003176                  MOV     #SUMLO1,-(SP)   ;PUT POINTER ON STACK
 7419  031762  004737  053366                  JSR     PC,$DB2O        ;CONVERT TO OCTAL
 7420  031766  004737  053702                  JSR     PC,$SUPRS       ;TYPE IT
 7421  031772  104401  001315                  TYPE    ,$CRLF          ;TYPE <CR>,<LF>
 7422  031776  062766  000002  000014          ADD     #2,14(SP)       ;INCREMENT R1 INDEX
 7423  032004  062716  000002          28$:    ADD     #2,(SP) ;GET NORMAL RETURN PC
 7424  032010  104410                  30$:    RESREG                  ;RESTORE R0-R5
 7425  032012  000207                          RTS     PC              ;RETURN
 7426
 7427
 7428
 7429                                  ;;*****************************************************************
 7430                                  ;*MXWRDC - GET MAX WORD COUNT FOR CURRENT MEM LIMITS AND CURRENT MA,
 7431                                  ;*AND LEAVE THE WORD COUNT IN R1-R0. NO REGISTERS ARE SAVED.
 7432                                  ;;*****************************************************************
 7433  032014  013700  006126          MXWRDC: MOV     MAHILM,R0       ;GET LO BITS OF MEM LIM
 7434  032020  013701  006130                  MOV     MAHILM+2,R1     ;HI BITS OF MEM LIM
 7435  032024  163700  005762                  SUB     MA,R0           ;SUBTRACT MA FROM MAHILM
 7436  032030  005601                          SBC     R1
 7437  032032  163701  005764                  SUB     MA+2,R1
 7438  032036  100413                          BMI     27$             ;IF NEGATIVE, RETURN
 7439  032040  000241                          CLC                     ;DIVIDE BY 2 TO GET WORDS
 7440  032042  006001                          ROR     R1
 7441  032044  006000                          ROR     R0
 7442  032046  062700  000001                  ADD     #1,R0           ;INCREMENT BY 1 WORD
 7443  032052  005501                          ADC     R1
 7444  032054  005701                          TST     R1
 7445  032056  001403                          BEQ     27$             ;BR IF WORD COUNT < 65,536 DEC
 7446  032060  005000                          CLR     R0              ;MAKE WORD COUNT = 65,536
 7447  032062  012701  000001                  MOV     #1,R1
 7448  032066  000207          27$:    RTS     PC              ;RETURN
 7449
 7450
 7451
 7452                                  ;;*****************************************************************
 7453                                  .SBTTL  DRVSER - TYPE DRIVE SERIAL NUMBER (LOW 3 DIGITS)
 7454                                  ;*THIS SUBROUTINE TYPES 'DRIVE SER. NO.  XXX' (IN DECIMAL), WITH LEADING
 7455                                  ;*ZEROS SUPPRESSED.
 7456                                  ;;*****************************************************************
 7457  032070  104407          DRVSER: SAVREG                  ;SAVE R0-R5
 7458  032072  004737  027450                  JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
 7459  032076  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;SET READ STATUS COMMAND
 7460  032104  004737  040252                  JSR     PC,DRVCAL       ;READ STATUS OF THIS DRIVE
 7461  032110  104401  011335                  TYPE    ,DRIV           ;TYPE 'DRIVE'
 7462  032114  104401  011351                  TYPE    ,SERNM          ;TYPE '' SER. NO.  ''
 7463  032120  016501  000054                  MOV     P.A11(R5),R1    ;GET ''A'' STATUS BYTE 11
 7464  032124  012704  053470                  MOV     #$OCTVL,R4      ;GET ADDR OF CHAR BUFFER
 7465  032130  010446                          MOV     R4,-(SP)        ;STORE IT ON STACK FOR $SUPRS
 7466  032132  012703  000003                  MOV     #3,R3           ;INIT CHAR COUNT
```

```
 7467  032136  006101                        ROL     R1              ;INITIALIZE BIT POSITIONS
 7468  032140  006101                        ROL     R1
 7469  032142  006101                4$:     ROL     R1              ;GET NEXT 4 BITS
 7470  032144  006101                        ROL     R1
 7471  032146  006101                        ROL     R1
 7472  032150  006101                        ROL     R1
 7473  032152  010100                        MOV     R1,R0           ;GET A WORKING COPY
 7474  032154  042700  177760              BIC     #177760,R0      ;CLEAR ALL BUT LOW 4 BITS
 7475  032160  052700  000060              BIS     #'0,R0          ;CONVERT A DIGIT TO ASCII
 7476  032164  110024                        MOVB    R0,(R4)+        ;PUT ASCII DIGIT INTO CHAR BUFFER
 7477  032166  005303                        DEC     R3              ;DECREMENT CHAR COUNT
 7478  032170  001364                        BNE     4$              ;BR IF NOT 3 CHARS YET
 7479  032172  105014                        CLRB    (R4)            ;INSERT NULL TERMINATOR
 7480  032174  004737  053702              JSR     PC,@#$SUPRS     ;TYPE DRIVE SER. NUMBER
 7481  032200  104401  001315              TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7482  032204  104410                        RESREG                  ;RESTORE R0-R5
 7483  032206  000207                        RTS     PC              ;RETURN
 7484
 7485
 7486
 7487                                ;;*****************************************************************
 7488                                .SBTTL  CRTSER - TYPE CARTRIDGE SERIAL NUMBER
 7489                                ;*THIS SUBROUTINE TYPES "CART. SER. NO.  XXXXXXXXXX" (IN OCTAL),
 7490                                ;*WITH LEADING ZEROS SUPPRESSED.
 7491                                ;;*****************************************************************
 7492  032210  004737  027450        CRTSER: JSR     PC,INITSS       ;CLEAR S.S. AND PARAMETERS
 7493  032214  142765  000020  000007        BICB    #B.CFMT,P.CS1H(R5)      ;SET 22 SECTOR FORMAT
 7494  032222  105737  003125              TSTB    FORMAT          ;CHECK THE ACTUAL FORMAT
 7495  032226  001402                        BEQ     4$              ;BR IF 22 SECTORS
 7496  032230  105265  000004              INCB    P.SECT(R5)      ;IF 20 SECTORS, READ SECTOR 1
 7497  032234  112765  000121  000001  4$:     MOVB    #RDDATA,P.CMND(R5) ;SET READ COMMAND
 7498  032242  013765  017706  000002        MOV     LSTCYL,P.CYLN(R5) ;SET CYL = 632(OCT)/1456(OCT) FOR RK07
 7499  032250  112765  000002  000005        MOVB    #LSTTRK,P.TRCK(R5) ;SET TRACK = 2
 7500  032256  012765  063520  000010        MOV     #RWBUF,P.BALO(R5) ;SET READ BUFFER ADDRESS
 7501  032264  012765  177776  000012        MOV     #-2,P.WC(R5)    ;SET WORD COUNT TO READ 2 WORDS
 7502  032272  004737  040252              JSR     PC,DRVCAL       ;READ SERIAL NO. IN BSF
 7503  032276  104401  011342              TYPE    ,CART           ;TYPE   "CART."
 7504  032302  104401  011351              TYPE    ,SERNM          ;TYPE "SER. NO.  "
 7505  032306  012746  063520              MOV     #RWBUF,-(SP)    ;GET POINTER FOR $DB2O
 7506  032312  004737  053366              JSR     PC,@#$DB2O      ;CONVERT BINARY TO OCTAL
 7507  032316  004737  053702              JSR     PC,@#$SUPRS     ;TYPE CART. SERIAL NO. IN OCTAL
 7508  032322  104401  001315              TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7509  032326  000207                        RTS     PC              ;RETURN
 7510
 7511
 7512
 7513                                ;;*****************************************************************
 7514                                .SBTTL  SEEKER - PERFORM IMPLICIT OR EXPLICIT SEEK
 7515                                ;*THIS SUBROUTINE CHECKS SWR BIT 7, AND IF IT IS 0, AN IMPLICIT
 7516                                ;*SEEK VIA THE READ HEADER COMMAND IS PERFORMED. IF THE HEADER
 7517                                ;*DOES NOT CONTAIN THE CORRECT CYLINDER NO. AN ERROR IS REPORTED.
 7518                                ;*IF SWR BIT 7 = 1, AN EXPLICIT SEEK COMMAND IS PERFORMED.
 7519                                ;*ON ENTRY TO THE SUBROUTINE, THE DRIVE AND CYLINDER NUMBERS
 7520                                ;*MUST BE PRE-LOADED INTO THE PARAMETER BLOCK.
 7521                                ;* CALL -    JSR     PC,SEEKER
 7522                                ;;*****************************************************************
```

```
7523
7524  032330  010046                    SEEKER: MOV     R0,-(SP)          ;SAVE R0
7525  032332  112765  000117  000001            MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
7526  032340  032777  000200  146572            BIT     #BIT07,@SWR       ;SEE IF EXPLICIT SEEKS REQUESTED
7527  032346  001023                            BNE     2$                ;BR IF EXPLICIT SEEKS
7528  032350  112765  000125  000001            MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
7529  032356  004737  040252                    JSR     PC,DRVCAL         ;READ A HEADER
7530  032362  016200  000024                    MOV     RKDB(R2),R0       ;GET WORD 1 OF ACTUAL HEADER
7531  032366  020065  000002                    CMP     R0,P.CYLN(R5)     ;COMPARE TO EXPECTED CYLINDER
7532  032372  001413                            BEQ     4$                ;BR IF EQUAL
7533  032374  004737  041650                    JSR     PC,REPSUP         ;STORE PREV AND CURRENT CMNDS
7534  032400  016537  000002  001174            MOV     P.CYLN(R5),$REG5 ;GET GOOD CYL NO.
7535  032406  010037  001176                    MOV     R0,$REG6          ;GET BAD CYL NO.
7536  032412  104062                            ERROR   62                ;CYLINDER MISCOMPARE
7537  032414  000402                            BR      4$
7538  032416  004737  040252          2$:       JSR     PC,DRVCAL         ;PERFORM COMMAND
7539  032422  012600                4$:         MOV     (SP)+,R0          ;RESTORE R0
7540  032424  000207                            RTS     PC                ;RETURN
7541
7542
7543
7544                                ;;*****************************************************************
7545                                .SBTTL  STALL - STALL FOR ST UNIT STALL TIMES
7546                                ;*IF SWR BIT 8 = 0, THIS SUBROUTINE STALLS FOR ST STALL TIMES,
7547                                ;*WHERE A STALL TIME = 40 US, FOR AN "AVERAGE" CPU. IF BIT 8
7548                                ;*IS EQUAL TO 1, A RANDOM STALL IS APPLIED.
7549                                ;* CALL -   JSR     PC,STALL
7550                                ;;*****************************************************************
7551
7552  032426  010046                    STALL:  MOV     R0,-(SP)          ;SAVE R0
7553  032430  010146                            MOV     R1,-(SP)          ;SAVE R1
7554  032432  032777  000400  146500            BIT     #BIT08,@SWR       ;APPLY RANDOM STALL ?
7555  032440  001407                            BEQ     1$                ;BR IF NOT RANDOM
7556  032442  004737  052564                    JSR     PC,$RAND          ;GENERATE PSEUDO-RANDOM NUMBER
7557  032446  013700  052664                    MOV     $LONUM,R0         ;GET IT INTO R0
7558  032452  006200                            ASR     R0                ;SCALE IT DOWN
7559  032454  006200                            ASR     R0
7560  032456  000403                            BR      2$                ;GO STALL WITH RANDOM NO.
7561  032460  013700  005512          1$:       MOV     STALLS,R0         ;GET REQUESTED NO. OF STALLS
7562  032464  001406                            BEQ     6$                ;RETURN IF NO STALL REQUIRED
7563  032466  012701  000016          2$:       MOV     #14.,R1           ;SET CONSTANT FOR 40 US
7564  032472  005301                4$:         DEC     R1                ;INNER LOOP COUNTER
7565  032474  001376                            BNE     4$                ;INNER LOOP BR
7566  032476  005300                            DEC     R0                ;OUTER LOOP COUNTER
7567  032500  001372                            BNE     2$                ;OUTER LOOP BR
7568  032502  012601                6$:         MOV     (SP)+,R1          ;RESTORE R1
7569  032504  012600                            MOV     (SP)+,R0          ;RESTORE R0
7570  032506  000207                            RTS     PC                ;RETURN
7571
7572
7573
7574                                ;;*****************************************************************
7575                                .SBTTL  RNDADR - RANDOM CYLINDER ADDRESS GENERATOR
7576                                ;*THIS SUBROUTINE GENERATES A PSEUDO-RANDOM 9-BIT CYLINDER
7577                                ;*ADDRESS, AND LEAVES IT IN "CYLNDR". IT REQUIRES THE SYSMAC
7578                                ;*SUBROUTINE $RAND.
```

```
                                        ;* CALL -   JSR     PC,RNDADR
                                        ;;*****************************************************************
032510  004737  052564                  RNDADR: JSR     PC,$RAND        ;GENERATE 2 16-BIT RANDOM NUMBERS
032514  013737  052664  005514                  MOV     $LONUM,CYLNDR   ;GET A RANDOM NUMBER
032522  042737  177000  005514                  BIC     #177000,CYLNDR  ;SCALE IT TO 9 BITS
032530  023737  017706  005514                  CMP     LSTCYL,CYLNDR   ;SEE IF CYL IS TOO BIG
032536  002003                                  BGE     2$              ;BR IF CYL IS OK
032540  043737  017712  005514                  BIC     HOLD1,CYLNDR    ;SCALE DOWN TO A VALID CYLINDER
032546  000207                          2$:     RTS     PC              ;RETURN


                                        .SBTTL  ROUTINES TO HANDLE KW11-L OR P CLOCK

                                        ;;*****************************************************************
                                        ;*CLKON - ENABLE KW11-L OR P CLOCK INTERRUPT
                                        ;;*****************************************************************
032550  105737  003132                  CLKON:  TSTB    PCLKF           ;SEE WHICH CLOCK WILL BE USED
032554  001004                                  BNE     1$              ;BR IF P-CLOCK TO BE USED
032556  012777  000100  150364                  MOV     #100,@LKS       ;L-CLOCK, ENABLE INTERRUPT
032564  000207                                  RTS     PC              ;RETURN
032566  012777  174575  150360          1$:     MOV     #-1667.,@PKSB   ;SET 60 HZ. COUNT
032574  005737  000166                          TST     HZ              ;DETERMINE LINE FREQUENCY
032600  001403                                  BEQ     3$              ;BR IF 60 HZ.
032602  012777  174060  150344                  MOV     #-2000.,@PKSB   ;SET 50 HZ. COUNT
032610  012777  000131  150334          3$:     MOV     #131,@PKS       ;ENABLE INTERRUPT,CNT UP,REP. INT.
                                                                        ;  100 KHZ.,AND RUN
032616  000207                                  RTS     PC              ;RETURN

                                        ;;*****************************************************************
                                        ;*CLOCK - HANDLE KW11-L OR P CLOCK INTERRUPT
                                        ;;*****************************************************************
032620  005200                          CLOCK:  INC     R0              ;SIGNIFY A CLOCK TICK
032622  000002                                  RTI                     ;RETURN FROM INTR

                                        ;;*****************************************************************
                                        ;*CLKOF - DISABLE KW11-L OR P CLOCK INTERRUPT
                                        ;;*****************************************************************
032624  105737  003132                  CLKOF:  TSTB    PCLKF           ;SEE WHICH CLOCK USED
032630  001003                                  BNE     1$              ;BR IF P-CLOCK USED
032632  005077  150312                          CLR     @LKS            ;DISABLE KW11-L INTR
032636  000207                                  RTS     PC              ;RETURN
032640  005077  150306          1$:     CLR     @PKS            ;DISABLE KW11-P INTR
032644  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>
                        ;;*****************************************************************

032646  104407          CALBRT: SAVREG                  ;SAVE R0-R5
032650  005002                  CLR     R2              ;INIT CALIBRATION LOOP CONSTANT
032652  005037  003176          CLR     SUMLO1          ;INIT SUM TO 0
032656  005037  003200          CLR     SUMHI1
032662  005001          2$:     CLR     R1              ;INIT TIME-OUT INDICATOR
032664  005000                  CLR     R0              ;INIT CLOCK INTR INDIC.
                        ;GET IN SYNCH WITH CLOCK
032666  004737  032550          JSR     PC,CLKON        ;ENABLE KW11-L OR P CLOCK INTR
032672  005700          4$:     TST     R0              ;SEE IF CLOCK INTR REC'D YET
032674  001013                  BNE     8$              ;BR IF INTR REC'D
032676  005201                  INC     R1              ;INCREMENT CLOCK TIME-OUT INDICATOR
032700  001374                  BNE     4$              ;BR IF NO TIME-OUT
032702  104401  011517  6$:     TYPE    ,CLKFAL         ;TIMED-OUT ON KW11-L OR P INTERRUPT
032706  104401  011454          TYPE    ,TIMSKP         ;SAY ALL TIMING TESTS WILL BE SKIPPED
032712  105037  003133          CLRB    DOTIM           ;DON'T ALLOW TIMING TESTS
032716  017616  000000          MOV     @(SP),(SP)      ;FIX UP ERROR RETURN PC
032722  000464                  BR      18$             ;GO TO RETURN
                        ;RUN TIMER TO NEXT CLOCK TICK
032724  005000          8$:     CLR     R0              ;INIT CLOCK INTR INDIC
032726  005004                  CLR     R4              ;INIT HI BITS OF CYCLE COUNT
032730  005005                  CLR     R5              ;LO BITS
032732  004737  033104          JSR     PC,TIMER        ;COUNT CYCLES TIL NEXT CLOCK INTR
032736  032702                  6$                      ;ERROR RETURN ADDRESS
                        ;ADD MEASURED CYCLES TO SUM
032740  060537  003176          ADD     R5,SUMLO1       ;ADD MEASUREMENT TO SUM
032744  005537  003200          ADC     SUMHI1
032750  060437  003200          ADD     R4,SUMHI1
032754  103752                  BCS     6$              ;BR IF SUM OVERFLOW
032756  005202                  INC     R2              ;INCR CALIBRATION LOOP COUNT
032760  022702  000200          CMP     #128.,R2        ;SEE IF 128 MEASUREMENTS MADE YET
032764  001336                  BNE     2$              ;BR IF NOT YET
                        ;TAKE AVERAGE OF 128 MEASUREMENTS
032766  012702  000007          MOV     #7,R2           ;SET DIVIDE LOOP COUNT
032772  000241          12$:    CLC
032774  006037  003200          ROR     SUMHI1          ;DIVIDE SUM BY 2
033000  006037  003176          ROR     SUMLO1
033004  005302                  DEC     R2              ;DECREMENTT LOOP COUNT
033006  001371                  BNE     12$             ;BR IF DIVISION NOT DONE YET
                        ;DIVIDE LINE PERIOD IN N-SEC BY TIMER CYCLE COUNT TO GET CALIB. CONST.
033010  005000                  CLR     R0              ;SET DIVIDEND = 16666667 NS
033012  005001                  CLR     R1
033014  012702  000376          MOV     #376,R2
033020  012703  050053          MOV     #^D20523,R3
033024  005737  000166          TST     HZ              ;DETERMINE THE LINE FREQUENCY
033030  001404                  BEQ     16$             ;BR IF 60 HZ.
033032  012702  000461          MOV     #461,R2         ;SET DIVIDEND = 20000000 NS
033036  012703  026400          MOV     #^D11520,R3
033042  013704  003200  16$:    MOV     SUMHI1,R4       ;SET DIVISOR
033046  013705  003176          MOV     SUMLO1,R5
033052  004737  053210          JSR     PC,M.DPID       ;PERFORM THE DIVISION
033056  010237  003166          MOV     R2,TCONHI       ;GET FINAL TIMING CONSTANT
```

```
7691  033062  001307                  BNE     6$              ;BR IF HI BITS NOT 0
7692  033064  010337  003164          MOV     R3,TCONLO       ;GET LO BITS
7693  033070  062716  000002          ADD     #2,(SP)         ;FIX UP ERROR-FREE RETURN PC
7694  033074  004737  032624  18$:    JSR     PC,CLKOF        ;DISABLE KW11-L OR P CLOCK INTERRUPT
7695  033100  104410                  RESREG                  ;RESTORE R0-R5
7696  033102  000207                  RTS     PC              ;RETURN
7697
7698
7699
7700                                  ;;*****************************************************************
7701                                  .SBTTL  TIMER - SOFTWARE TIMER SUBROUTINE
7702                                  ;*THIS SUBROUTINE LOOPS THROUGH THE SOFTWARE TIMING LOOP,
7703                                  ;*INCREMENTING THE CYCLE COUNT IN R4-R5 EACH LOOP, AND CHECKS
7704                                  ;*THE INTERRUPT INDICATION IN R0, FOR COMPLETION. IF A COUNT
7705                                  ;*OF 4,194,304 IS REACHED IN R4-R5, A TIME-OUT ERROR RETURN
7706                                  ;*IS MADE TO THE ADDRESS LISTED AFTER THE CALL .
7707                                  ;* CALL -     S     PC,TIMER
7708                                  ;*          ,OR RETURN ADDRESS>
7709                                  ;;*************  ***************************************************
7710                                  ;*** SPECIAL TIMING LOOP ***
7711  033104  062705  000001  TIMER:  ADD     #1,R5           ;INCREMENT LO CYCLE COUNT
7712  033110  005504                  ADC     R4              ;ADD CARRY TO HI CYCLE COUNT
7713  033112  032704  000200          BIT     #BIT07,R4       ;SEE IF TIMED-OUT WAITING FOR INTR
7714  033116  001005                  BNE     4$              ;BR IF TIME-OUT
7715  033120  005700                  TST     R0              ;SEE IF INTERRUPT RECEIVED
7716  033122  001770                  BEQ     TIMER           ;BR IF INTERRUPT NOT REC'D YET
7717                                  ;*****************************
7718  033124  062716  000002          ADD     #2,(SP)         ;FIX UP ERROR-FREE RETURN PC
7719  033130  000207                  RTS     PC              ;ERROR-FREE RETURN
7720  033132  017616  000000  4$:     MOV     @(SP),(SP)      ;FIX UP ERROR RETURN PC
7721  033136  000207                  RTS     PC              ;ERROR RETURN
7722
7723
7724
7725                                  ;;*****************************************************************
7726                                  ;* INIVRB - INITIALIZE VARIABLES USED IN TIMING TESTS
7727                                  ;;*****************************************************************
7728  033140  005037  003170  INIVRB: CLR     BLWMIN          ;COUNT OF TIMES BELOW SPEC'D MIN
7729  033144  005037  003172          CLR     ABVMX1          ;COUNT OF FORWARD TIMES ABOVE SPEC'D MAX
7730  033150  005037  003174          CLR     ABVMX2          ;COUNT OF REVERSE TIMES ABOVE SPEC'D MAX
7731  033154  005037  003176          CLR     SUMLO1          ;FORWARD TIME SUM AND AVERAGE
7732  033160  005037  003200          CLR     SUMHI1
7733  033164  005037  003202          CLR     SUMLO2          ;REVERSE TIME SUM AND AVERAGE
7734  033170  005037  003204          CLR     SUMHI2
7735  033174  012737  177777  003212  MOV     #-1,MINIL1      ;MIN MEAS'D FORWARD TIME
7736  033202  012737  177777  003214  MOV     #-1,MINIH1
7737  033210  005037  003216          CLR     MAXIL1          ;MAX MEAS'D FORWARD TIME
7738  033214  005037  003220          CLR     MAXIH1
7739  033220  012737  177777  003222  MOV     #-1,MINIL2      ;MIN MEAS'D REVERSE TIME
7740  033226  012737  177777  003224  MOV     #-1,MINIH2
7741  033234  005037  003226          CLR     MAXIL2          ;MAX MEAS'D REVERSE TIME
7742  033240  005037  003230          CLR     MAXIH2
7743  033244  000207                  RTS     PC              ;RETURN
7744
7745
7746
```

```
                                        ;;*****************************************************************
                                        .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>
                                        ;;*****************************************************************
033246  010246                  TIMSEK: MOV     R2,-(SP)        ;SAVE R2
033250  010546                          MOV     R5,-(SP)        ;SAVE R5
033252  005000                          CLR     R0              ;INIT DRIVE INTR INDICATOR
033254  005004                          CLR     R4              ;INIT HI CYCLE COUNT
033256  012777  033426  147554          MOV     #SEEKHD,@RKVEC  ;SET SPECIAL RK06 SEEK TIMER HANDLER VECTOR
033264  112765  000117  000001          MOVB    #SEEK,P.CMND(R5) ;SET SEEK COMMAND
033272  010537  033314                  MOV     R5,2$           ;SET PARAM BLK ADDRESS FOR DRIVER
033276  005005                          CLR     R5              ;INIT LO CYCLE COUNT
033300  062702  000000                  ADD     #RKCS1,R2       ;GET ADR OF RKCS1 IN R2 FOR SEEKHD
033304  004737  040542                  JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
033310  004737  050534                  JSR     PC,C.INIT       ;START THE SEEK
033314  000000                  2$:     .WORD   0               ;PARAM BLK ADR GOES HERE
033316  004737  033104                  JSR     PC,TIMER        ;MEASURE TIME TIL DRIVE ATT'N
033322  033372                          6$                      ;ERROR RETURN ADDRESS FOR TIMER
033324  004737  034072                  JSR     PC,CNVTIM       ;CONVERT MEAS'D CYCLES TO TIME
033330  033372                          6$                      ;ERROR RETURN ADR FOR CNVTIM
033332  010200                          MOV     R2,R0           ;PUT HI TIME BITS IN R0
033334  010301                          MOV     R3,R1           ;PUT LO TIME BITS IN R1
033336  062766  000002  000004          ADD     #2,4(SP)        ;FIX UP ERROR-FREE RETURN PC
033344  012777  045234  147466  4$:     MOV     #I.INTR,@RKVEC  ;RESTORE RK06 INTR HANDLER
033352  012605                          MOV     (SP)+,R5        ;RESTORE R5
033354  012602                          MOV     (SP)+,R2        ;RESTORE R2
033356  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5) ;SET SUBSYSTEM CLEAR COMMAND
033364  004737  040252                  JSR     PC,DRVCAL       ;CLEAR THE SUBSYSTEM
033370  000207                          RTS     PC              ;RETURN
033372  013702  003036          6$:     MOV     RKBAS,R2        ;GET RK06 REG ADDR
033376  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;DISABLE RK06 INTERRUPT
033404  004737  041650                  JSR     PC,REPSUP       ;GET PREV AND CURRENT CMNDS
033410  004737  044316                  JSR     PC,TOPROC       ;GATHER STATUS
033414  104106                          ERROR   106             ;TIMED-OUT ON SEEK
033416  017666  000004  000004          MOV     @4(SP),4(SP)    ;FIX ERROR RETURN PC
033424  000747                          BR      4$              ;BR TO EXIT

                                        ;;*****************************************************************
                                        ;*SPECIAL INTERRUPT HANDLER FOR SEEKS IN SEEK TIMING TESTS
                                        ;;*****************************************************************
033426  011203                  SEEKHD: MOV     (R2),R3         ;GET BITS OF RKCS1
033430  100002                          BPL     2$              ;BR IF CERR NOT SET
033432  000137  045234                  JMP     I.INTR          ;LET DRIVER PROCESS THE ERROR
033436  032703  040000          2$:     BIT     #DI,R3          ;SEE IF DRIVE INTR SET YET
033442  001401                          BEQ     4$              ;BR IF NOT YET
033444  005200                          INC     R0              ;SET RK06 INTR INDICATOR
033446  000002                  4$:     RTI                     ;RETURN FROM INTERRUPT


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

```
                                ;*FDTERM - COMPUTE A TERM OF THE FORWARD SEEK AVERAGE FORMULA,
                                ;*AND ADD IT TO THE FORWARD SUM IN SUMLO1-SUMHI1-SUMLO2-SUMHI2.
                                ;;*****************************************************************
033450  104407                  FDTERM: SAVREG                  ;SAVE R0-R5
033452  010002                          MOV     R0,R2           ;GET MEASUREMENT INTO  R2-R3
033454  010103                          MOV     R1,R3
033456  005004                          CLR     R4
033460  013705  005542                  MOV     SCRACH,R5       ;GET COEFFICIENT
033464  006305                          ASL     R5              ;MULTIPLY BY 2
033466  004737  053150                  JSR     PC,M.DPIM       ;MULTIPLY COEFFICIENT
033472  060337  003204                  ADD     R3,SUMHI2       ;ADD TO FORWARD SUM
033476  005502                          ADC     R2
033500  060237  003202                  ADD     R2,SUMLO2
033504  005537  003200                  ADC     SUMHI1
033510  104410                          RESREG                  ;RESTORE R0-R5
033512  000207                          RTS     PC


                                ;;*****************************************************************
                                ;*RVTERM - COMPUTE  A TERM  OF THE REVERSE AVERAGE FORMULA,
                                ;*AND ADD IT TO THE REVERSE SUM IN MAXIL1-MAXIH1-MAXIL2-MAXIH2.
                                ;;*****************************************************************
033514  104407                  RVTERM: SAVREG           ;SAVE R0-R5
033516  010002                          MOV     R0,R2           ;GET MEASUREMENT INTO R2-R3
033520  010103                          MOV     R1,R3
033522  005004                          CLR     R4
033524  013705  005542                  MOV     SCRACH,R5       ;GET COEFFICIENT
033530  006305                          ASL     R5              ;MULTIPLY BY 2
033532  004737  053150                  JSR     PC,M.DPIM       ;MULTIPLY COEFFICIENT
033536  060337  003230                  ADD     R3,MAXIH2       ;ADD TO REVERSE SUM
033542  005502                          ADC     R2
033544  060237  003226                  ADD     R2,MAXIL2
033550  005537  003220                  ADC     MAXIH1
033554  104410                          RESREG                  ;RESTORE R0-R5
033556  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.
                        ;;*****************************************************************
033560  020063  000002  CMPTIM: CMP     R0,2(R3)        ;COMPARE HI BITS TO HI MIN
033564  101006                  BHI     6$              ;BR IF > MIN
033566  103402                  BLO     4$              ;BR IF < MIN
033570  020113                  CMP     R1,(R3)         ;COMPARE LO BITS TO LO MIN
033572  103003                  BHIS    6$              ;BR IF > OR = MIN
033574  010063  000002  4$:     MOV     R0,2(R3)        ;SET NEW MIN HI BITS
033600  010113                  MOV     R1,(R3)         ;SET NEW MIN LO BITS
033602  020063  000006  6$:     CMP     R0,6(R3)        ;COMPARE HI BITS TO HI MAX
033606  103410                  BLO     10$             ;BR IF < MAX
033610  101003                  BHI     8$              ;BR IF > MAX
033612  020163  000004          CMP     R1,4(R3)        ;COMPARE LO BITS TO LO MAX
```

```
7859  033616  101404                  BLOS    10$             ;BR IF < OR = MAX
7860  033620  010063  000006    8$:   MOV     R0,6(R3)        ;SET NEW MAX HI BITS
7861  033624  010163  000004          MOV     R1,4(R3)        ;SET NEW MAX LO BITS
7862  033630  000207            10$:  RTS     PC              ;RETURN
7863
7864
7865
7866                              ;;*****************************************************************
7867                              ;*GETAVG - COMPUTE FORWARD AND REVERSE SEEK TIME AVERAGES BY
7868                              ;*DIVIDING TIME SUMS BY NO. OF MEASUREMENTS (PASSED IN R3 ON ENTRY).
7869                              ;;*****************************************************************
7870  033632  104407              GETAVG: SAVREG                  ;SAVE R0-R5
7871  033634  010305                      MOV     R3,R5           ;LO DIVISOR = MEASUREMENT COUNT
7872  033636  005000                      CLR     R0              ;SET DIVIDEND = FORWARD TIME SUM
7873  033640  005001                      CLR     R1
7874  033642  013702  003200              MOV     SUMHI1,R2
7875  033646  013703  003176              MOV     SUMLO1,R3
7876  033652  005004                      CLR     R4              ;HI DIVISOR = 0
7877  033654  004737  053210              JSR     PC,M.DPID       ;PERFORM DIVISION
7878  033660  010237  003200              MOV     R2,SUMHI1       ;STORE FORWARD TIME AVG
7879  033664  010337  003176              MOV     R3,SUMLO1
7880  033670  005000                      CLR     R0              ;SET DIVIDEND = REVERSE TIME SUM
7881  033672  005001                      CLR     R1
7882  033674  013702  003204              MOV     SUMHI2,R2
7883  033700  013703  003202              MOV     SUMLO2,R3
7884  033704  004737  053210              JSR     PC,M.DPID       ;PERFORM DIVISION
7885  033710  010237  003204              MOV     R2,SUMHI2       ;STORE REVERSE TIME AVG
7886  033714  010337  003202              MOV     R3,SUMLO2
7887  033720  104410                      RESREG                  ;RESTORE R0-R5
7888  033722  000207                      RTS     PC              ;RETURN
7889
7890
7891
7892                              ;;*****************************************************************
7893                              .SBTTL  TYPTMS - TYPE RESULTS OF SEEK TIMING MEASUREMENTS
7894                              ;*THIS SUBROUTINE TYPES THE MIN,MAX, AND AVG SEEK TIMES
7895                              ;*(IN DECIMAL) MEASURED IN ONE OF THE SEEK TIMING TESTS,
7896                              ;*IN BOTH THE FORWARD AND REVERSE DIRECTIONS. ALSO TYPED ARE
7897                              ;*THE MAX SPEC'D TIME AND THE NO. OF SEEKS EXCEEDING IT,
7898                              ;* AND THE TOTAL NO. OF MEASUREMENTS. POINTER TO SPECIFIED
7899                              ;*MAX MESSAGE MUST BE IN R3 ON ENTRY.
7900                              ;;*****************************************************************
7901
7902  033724                      TYPTMS:
7903                              ;TYPE FORWARD MEASUREMENTS
7904  033724  104401  011663              TYPE    ,FORWRD         ;TYPE "***FORWARD DIRECTION***"
7905  033730  012701  003212              MOV     #MINIL1,R1      ;POINTER TO LO MIN
7906  033734  004737  034144              JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
7907  033740  104401  001315              TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
7908  033744  012701  003216              MOV     #MAXIL1,R1      ;POINTER TO LO MAX
7909  033750  004737  034170              JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
7910  033754  013746  003172              MOV     ABVMX1,-(SP)    ;GET COUNT OF TIMES ABOVE MAX
7911  033760  104405                      TYPDS                   ;CONVERT AND TYPE IT
7912  033762  010337  033770              MOV     R3,4$           ;SET POINTER TO SPEC'D MAX MESSAGE
7913  033766  104401                      TYPE                    ;TYPE " OF XXX ABOVE MAX OF XXXXX US "
7914  033770  000000            4$:     .WORD   0               ;SPEC'D MAX MSG POINTER GOES HERE
```

```
 7915  033772  012701  003176                MOV     #SUMLO1,R1      ;SET POINTER TO LO AVG
 7916  033776  004737  034214                JSR     PC,TYPAVG       ;TYPE AVG TIME
 7917  034002  104401  001315                TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7918                                ;TYPE REVERSE MEASUREMENTS
 7919  034006  104401  011701                TYPE    ,REVRSE         ;TYPE "***REVERSE DIRECTION***"
 7920  034012  012701  003222                MOV     #MINIL2,R1      ;POINTER TO LO MIN
 7921  034016  004737  034144                JSR     PC,TYPMIN       ;TYPE MIN MEAS'D TIME
 7922  034022  104401  001315                TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7923  034026  012701  003226                MOV     #MAXIL2,R1      ;POINTER TO LO MAX
 7924  034032  004737  034170                JSR     PC,TYPMAX       ;TYPE MAX MEAS'D TIME
 7925  034036  013746  003174                MOV     ABVMX2,-(SP)    ;GET COUNT OF TIMES ABOVE MAX
 7926  034042  104405                        TYPDS                   ;CONVERT AND TYPE IT
 7927  034044  010337  034052                MOV     R3,6$           ;GET POINTER TO SPEC'D MAX MESSAGE
 7928  034050  104401                        TYPE                    ;TYPE " OF XXX ABOVE MAX OF XXXXX US "
 7929  034052  000000            6$:         .WORD   0               ;SPEC'D MAX MSG POINTER GOES HERE
 7930  034054  012701  003202                MOV     #SUMLO2,R1      ;SET POINTER TO LO AVG
 7931  034060  004737  034214                JSR     PC,TYPAVG       ;TYPE AVG TIME
 7932  034064  104401  001315                TYPE    ,$CRLF          ;TYPE <CR> AND <LF>
 7933  034070  000207                        RTS     PC              ;RETURN
 7934
 7935
 7936
 7937                                ;*****************************************************************
 7938                                ;* CNVTIM - CONVERT SOFTWARE TIMER CYCLES TO TIME
 7939                                ;*THIS SUBROUTINE MULTIPLIES THE CYCLE COUNT IN R4-R5 BY THE
 7940                                ;*TIME CONSTANT IN TCONHI-TCONLO, TO OBTAIN TIME IN N-SEC. THIS
 7941                                ;*IS THEN DIVIDED BY 1000 (DEC) TO OBTAIN TIME IN MICRO-SEC.
 7942                                ;*THIS TIME IS RETURNED IN R2-R3. IF THE MULTIPLICATION OVERFLOWS
 7943                                ;*32 BITS, AN ERROR RETURN IS MADE TO THE ADDRESS LISTED AFTER THE
 7944                                ;*CALL TO CNVTIM.
 7945                                ;*CALL -    JSR    PC,CNVTIM
 7946                                ;*          <ERROR RETURN ADDRESS>
 7947                                ;;*****************************************************************
 7948  034072  013702  003166        CNVTIM: MOV     TCONHI,R2       ;SET MULTIPLIER = TIME CONST
 7949  034076  013703  003164                MOV     TCONLO,R3
 7950  034102  004737  053150                JSR     PC,M.DPIM       ;MULTIPLY
 7951  034106  005700                        TST     R0              ;MAKE SURE HI 2 WORDS ARE 0
 7952  034110  001002                        BNE     4$              ;BR IF NOT 0
 7953  034112  005701                        TST     R1
 7954  034114  001403                        BEQ     6$
 7955  034116  017616  000000        4$:     MOV     @(SP),(SP)      ;SET ERROR RETURN PC
 7956  034122  000207                        RTS     PC              ;ERROR RETURN
 7957  034124  005004                6$:     CLR     R4              ;SET HI BITS = 0
 7958  034126  012705  001750                MOV     #1000.,R5       ;SET LO DIVISOR = 1000 (DEC)
 7959  034132  004737  053210                JSR     PC,M.DPID       ;CONVERT TIME TO US
 7960  034136  062716  000002                ADD     #2,(SP)         ;FIX ERROR-FREE RETURN PC
 7961  034142  000207                        RTS     PC              ;ERROR-FREE RETURN
 7962
 7963
 7964
 7965                                ;;*****************************************************************
 7966                                ;*TYPMIN - TYPE MIN MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
 7967                                ;*ON ENTRY.
 7968                                ;;*****************************************************************
 7969  034144  104401  011717        TYPMIN: TYPE    ,MINEQ          ;TYPE 'MIN = "
 7970  034150  010146                        MOV     R1,-(SP)        ;GET POINTER TO LO TIME
```

```
7971  034152  004737  053506                  JSR     PC,@#$DB2D      ;CONVERT TO DEC
7972  034156  004737  053702                  JSR     PC,@#$SUPRS     ;TYPE IT
7973  034162  104401  011744                  TYPE    ,MICROS         ;TYPE 'US ''
7974  034166  000207                          RTS     PC
7975
7976
7977
7978                                  ;;*****************************************************************
7979                                  ;*TYPMAX - TYPE MAX MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
7980                                  ;*ON ENTRY.
7981                                  ;;*****************************************************************
7982  034170  104401  011726          TYPMAX: TYPE    ,MAXEQ          ;TYPE 'MAX = ''
7983  034174  010146                          MOV     R1,-(SP)        ;GET POINTER TO LO TIME
7984  034176  004737  053506                  JSR     PC,@#$DB2D      ;CONVERT TO DEC
7985  034202  004737  053702                  JSR     PC,@#$SUPRS     ;TYPE IT
7986  034206  104401  011744                  TYPE    ,MICROS         ;TYPE 'US ''
7987  034212  000207                          RTS     PC
7988
7989
7990
7991                                  ;;*****************************************************************
7992                                  ;*TYPAVG - TYPE AVERAGE MEAS'D TIME IN US. POINTER TO LO TIME IS IN R1
7993                                  ;*ON ENTRY.
7994                                  ;;*****************************************************************
7995  034214  104401  011735          TYPAVG: TYPE    ,AVGEQ          ;TYPE ''AVG = ''
7996  034220  010146                          MOV     R1,-(SP)        ;GET POINTER TO LO TIME
7997  034222  004737  053506                  JSR     PC,@#$DB2D      ;CONVERT TO DEC
7998  034226  004737  053702                  JSR     PC,@#$SUPRS     ;TYPE IT
7999  034232  104401  011744                  TYPE    ,MICROS         ;TYPE 'US ''
8000  034236  000207                          RTS     PC
8001
8002
8003
8004.
8005                                  ;;*****************************************************************
8006                                  ;* LODSEC - THIS SUBROUTINE LOADS THE CONTENTS OF R0 INTO ALL
8007                                  ;*256(DEC) WORDS OF THE DATA BUFFER (RWBUF).
8008                                  ;;*****************************************************************
8009  034240  104407                  LODSEC: SAVREG                  ;SAVE R0-R5
8010  034242  012701  063520                  MOV     #RWBUF,R1       ;GET ADDRESS OF DATA BUF INTO R1
8011  034246  010021                  2$:     MOV     R0,(R1)+        ;PUT WORD INTO BUFFER
8012  034250  020127  064520                  CMP     R1,#RWBUF+512.  ;SEE IF 256 WORDS WRITTEN YET
8013  034254  001374                          BNE     2$              ;BR IF NOT DONE YET
8014  034256  104410                          RESREG                  ;RESTORE R0-R5
8015  034260  000207                          RTS     PC              ;RETURN
8016
8017
8018
8019                                  ;;*****************************************************************
8020                                  ;* TRKCHK - THIS SUBROUTINE DOES A WRITE-CHECK OF SECTOR FS ON CYL
8021                                  ;*FC, FOR EACH OF TRACKS 0, 1, AND 2. THE DATA IS COMPARED TO THE
8022                                  ;*TRACK NUMBER + 100(OCT) FOR THAT PARTICULAR BLOCK OF DATA WRITTEN.
8023                                  ;*THE PARAMETER BLOCK MUST BE PRE-LOADED WITH THE PROPER PARAMETERS.
8024                                  ;;*****************************************************************
8025  034262  104407                  TRKCHK: SAVREG                  ;SAVE R0-R5
8026  034264  016546  000004                  MOV     P.SECT(R5),-(SP) ;SAVE TRACK AND SECTOR PARAMETERS
```

```
8027  034270  105065  000005                  CLRB    P.TRCK(R5)        ;CLEAR THE TRACK NO.
8028                                          ;LOAD THE DATA BUFFER WITH THE CURRENT TRACK NO.
8029  034274  116500  000005          2$:     MOVB    P.TRCK(R5),R0     ;GET TRACK NO. INTO R0
8030  034300  062700  000100                  ADD     #100,R0           ;ADD 100(OCT) TO TRACK NO.
8031  034304  004737  034240                  JSR     PC,LODSEC         ;LOAD DATA BUF WITH TRACK NO. + 100(OCT)
8032  034310  112765  000131  000001          MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
8033  034316  004737  040252                  JSR     PC,DRVCAL         ;PERFORM THE WRITE CHECK
8034  034322  105265  000005                  INCB    P.TRCK(R5)        ;INCREMENT THE TRACK NO.
8035  034326  122765  000003  000005          CMPB    #3,P.TRCK(R5)     ;SEE IF DONE WITH ALL TRACKS
8036  034334  001357                          BNE     2$                ;BR IF NOT DONE YET
8037  034336  012665  000004                  MOV     (SP)+,P.SECT(R5)  ;RESTORE TRACK AND SECTOR PARAMETERS
8038  034342  104410                          RESREG                    ;RESTORE R0-R5
8039  034344  000207                          RTS     PC                ;RETURN
8040
8041
8042
8043                                          ;;*****************************************************************
8044                                          ;*LODP14 - GENERATE SIXTEEN PSEUDO-RANDOM NUMBERS AND LOAD THEM
8045                                          ;*INTO THE PATTERN 14 TABLE.
8046                                          ;;*****************************************************************
8047  034346  104407                  LODP14: SAVREG                    ;SAVE R0-R5
8048  034350  012701  000010                  MOV     #8.,R1            ;INIT LOOP COUNTER TO 8.
8049  034354  012700  007116                  MOV     #PAT14,R0         ;GET ADDRESS OF PATTERN 14 BUFFER
8050  034360  004737  052564          4$:     JSR     PC,$RAND          ;GENERATE 2 16-BIT RANDOM NUMBERS
8051  034364  013720  052664                  MOV     $LONUM,(R0)+      ;PUT ONE NUMBER INTO PATTERN
8052  034370  013720  052662                  MOV     $HINUM,(R0)+      ;PUT OTHER NO. INTO PATTERN
8053  034374  005301                          DEC     R1                ;SEE IF 16 WORDS LOADED YET
8054  034376  001370                          BNE     4$                ;BR IF NOT YET
8055  034400  104410                          RESREG                    ;RESTORE R0-R5
8056  034402  000207                          RTS     PC                ;RETURN
8057
8058
8059
8060                                          ;;*****************************************************************
8061                                          .SBTTL  FINADR - COMPUTE FINAL PACK ADDRESS
8062                                          ;*THIS SUBROUTINE IS USED AFTER A DATA TRANSFER HAS COMPLETED, TO
8063                                          ;*COMPUTE THE FINAL PACK ADDRESS AT TERMINATION. THE 2'S COMP. OF THE
8064                                          ;*WORD NO. AT THE POINT OF INTEREST IS PASSED IN LASTWC. THIS IS USED WITH
8065                                          ;*P.CYLN(R5), P.TRCK(R5), AND P.SECT(R5) TO COMPUTE THE CORRESPONDING
8066                                          ;*PACK ADDRESS, WHICH IS RETURNED IN FINCYL, FINTRK, AND FINSEC.
8067                                          ;*THE SUBROUTINE IS USED TO DETERMINE THE PACK ADDRESS OF A SOFTWARE
8068                                          ;*DATA MISCOMPARE.
8069                                          ;;*****************************************************************
8070  034404  104407                  FINADR: SAVREG                    ;SAVE R0-R5
8071  034406  013746  005600                  MOV     @LASTWC,-(SP)     ;STORE WORD COUNT
8072  034412  005416                          NEG     @(SP)             ;MAKE IT POSITIVE
8073  034414  005046                  18$:    CLR     -(SP)             ;MAKE ROOM ON STACK
8074  034416  116616  000003                  MOVB    3(SP),(SP)        ;STORE NO. OF SECTORS TRANSFERRED
8075  034422  005066  000002                  CLR     2(SP)             ;CLEAR LOCATION ON STACK
8076  034426  116566  000004  000002          MOVB    P.SECT(R5),2(SP)  ;STORE STARTING SECTOR
8077  034434  066616  000002                  ADD     2(SP),(SP)        ;DETERMINE FINAL SECTOR ADDRESS
8078  034440  005066  000002                  CLR     2(SP)             ;CLEAR NO. OF TRACKS TRANSFERRED
8079  034444  012700  000026                  MOV     #22.,R0           ;SET FOR 22 SECTORS
8080  034450  105737  003125                  TSTB    FORMAT            ;DETERMINE THE FORMAT
8081  034454  001402                          BEQ     19$               ;BR IF 22 SECTORS
8082  034456  012700  000024                  MOV     #20.,R0           ;SET FOR 20 SECTORS
```

```
8083  034462  020016            19$:    CMP     R0,(SP)          ;CHECK FOR SECTOR OVERFLOW
8084  034464  101004                    BHI     20$              ;NO, CHECK IF SECTOR CORRECT
8085  034466  160016                    SUB     R0,(SP)          ;DECREMENT SECTOR COUNT BY 20 OR 22
8086  034470  005266  000002            INC     2(SP)            ;INCREMENT TRACKS TRANSFERRED
8087  034474  000772                    BR      19$              ;CHECK FOR SECTOR OVERFLOW
8088  034476  112637  005577    20$:    MOVB    (SP)+,FINSEC     ;STORE FINAL SECTOR
8089  034502  005046                    CLR     -(SP)            ;MAKE ROOM FOR TRACKS TRANSFERRED
8090  034504  116516  000005            MOVB    P.TRCK(R5),(SP)  ;STORE STARTING TRACK
8091  034510  066616  000002            ADD     2(SP),(SP)       ;DETERMINE FINAL TRACK ADDRESS
8092  034514  005066  000002            CLR     2(SP)            ;CLEAR FINAL CYLINDER
8093  034520  122716  000003    21$:    CMPB    #3,(SP)          ;CHECK FOR TRACK OVERFLOW
8094  034524  101005                    BHI     22$              ;NO, CHECK FINAL TRACK
8095  034526  162716  000003            SUB     #3,(SP)          ;DECREMENT TRACK COUNT BY 3
8096  034532  005266  000002            INC     2(SP)            ;INCR CYL COUNT
8097  034536  000770                    BR      21$              ;CHECK FOR TRACK OVERFLOW
8098  034540  112637  005576    22$:    MOVB    (SP)+,FINTRK     ;STORE FINAL TRACK
8099  034544  066516  000002            ADD     P.CYLN(R5),(SP)  ;CALCULATE FINAL CYLINDER
8100  034550  011637  005574            MOV     (SP),FINCYL      ;STORE FINAL  CYLINDER
8101  034554  005726                    TST     (SP)+            ;CLEAN OFF STACK
8102  034556  104410                    RESREG                   ;RESTORE R0-R5
8103  034560  000207                    RTS     PC               ;RETURN
8104
8105
8106
8107                                    ;;*****************************************************************
8108                                    .SBTTL  LODBUF - LOAD THE READ/WRITE DATA BUFFER
8109                                    ;*THIS SUBROUTINE LOADS THE READ/WRITE DATA BUFFER (POINTED TO BY
8110                                    ;* PMA AND PMA+2) WITH THE APPROPRIATE REPEATING DATA PATTERN. IF
8111                                    ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA WHICH IS LOADED IS COMPRISED
8112                                    ;*OF ALL THE PATTERNS 00-15 (QUICK VERIFY DEFAULT DATA TEST).
8113                                    ;*IF PATRN IS NOT 0, THE NO. OF WORDS IN THE WORD WDSXFR
8114                                    ;*OF THE PATTERN IDENTIFIED BY THE NO. OF THE BIT SET IN R1 ON
8115                                    ;*ENTRY, ARE LOADED INTO THE BUFFER.
8116                                    ;;*****************************************************************
8117
8118  034562  104407    LODBUF: SAVREG                           ;SAVE R0-R5
8119  034564  013702  005572            MOV     WDSXFR,R2        ;GET NO. OF WORDS
8120  034570  005737  005544            TST     PATRN            ;SEE IF QUICK VERIFY DATA TEST DESIRED
8121  034574  001007                    BNE     3$               ;BR IF NOT QUICK VERIFY
8122  034576  012700  006216            MOV     #PAT00,R0        ;SET DATA PATTERN STARTING ADDRESS
8123  034602  012746  000400            MOV     #256.,-(SP)      ;SET PATTERN WORD COUNT
8124  034606  012701  063520            MOV     #RWBUF,R1        ;SET BUFFER ADDRESS
8125  034612  000463                    BR      30$              ;PROCEED
8126  034614  005000            3$:     CLR     R0               ;INIT PATTERN NUMBER
8127  034616  032701  000001    4$:     BIT     #BIT0,R1         ;SEE IF THIS BIT IS SET
8128  034622  001003                    BNE     6$               ;BR IF THIS BIT IS SET
8129  034624  005200                    INC     R0               ;INCREMENT PATTERN NO.
8130  034626  006201                    ASR     R1               ;SHIFT TO EXAMINE NEXT BIT
8131  034630  000772                    BR      4$               ;BR TO CHECK NEXT BIT
8132  034632  006300            6$:     ASL     R0               ;MULTIPLY PATTERN NO. BY 32(DEC)
8133  034634  006300                    ASL     R0
8134  034636  006300                    ASL     R0
8135  034640  006300                    ASL     R0
8136  034642  006300                    ASL     R0
8137  034644  062700  006216            ADD     #PAT00,R0        ;GET ADDRESS OF DESIRED PATTERN
8138  034650  012746  000020            MOV     #16.,-(SP)       ;SET PATTERN WORD COUNT
```

```
8139  034654  013701  005602                  MOV     PMA,R1          ;SET BUFFER ADDRESS
8140  034660  005737  055662                  TST     $KT11           ;SEE IF MEM MGT PRESENT
8141  034664  100036                          BPL     30$             ;BR IF NOT PRESENT
8142                                  ;BUFFER LOAD LOOP FOR MEM MGT STARTS HERE
8143  034666  013737  003206  172354          MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
8144  034674  052737  000001  177572          BIS     #BIT0,@#SR0     ;TURN ON MEMORY MANAGEMENT
8145  034702  042701  160000                  BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
8146  034706  052701  140000                  BIS     #140000,R1
8147  034712  010003                  22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
8148  034714  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8149  034716  012321                  24$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
8150  034720  032701  020000                  BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
8151  034724  001405                          BEQ     26$             ;BR IF NO OVERFLOW
8152  034726  062737  000200  172354          ADD     #200,@#KIPAR6   ;INCREMENT PAR BY 4K FOR NEW PAGE
8153  034734  042701  020000                  BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
8154  034740  005302                  26$:    DEC     R2              ;DECREMENT WORD COUNTER
8155  034742  001403                          BEQ     28$             ;BR IF ALL DONE
8156  034744  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
8157  034746  001363                          BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
8158  034750  000760                          BR      22$             ;BR TO REPEAT THE PATTERN
8159  034752  042737  000001  177572  28$:    BIC     #BIT0,@#SR0     ;DISABLE MEMORY MANAGEMENT
8160  034760  000410                          BR      44$             ;GO TO RETURN
8161                                  ;BUFFER LOAD LOOP FOR NO MEM MGT STARTS HERE
8162  034762  010003                  30$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
8163  034764  011604                          MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8164  034766  012321                  34$:    MOV     (R3)+,(R1)+     ;LOAD A DATA WORD INTO BUFFER
8165  034770  005302                          DEC     R2              ;DECREMENT WORD COUNTER
8166  034772  001403                          BEQ     44$             ;BR IF ALL DONE
8167  034774  005304                          DEC     R4              ;DECREMENT PATTERN WORD COUNT
8168  034776  001373                          BNE     34$             ;BR IF NOT DONE WITH PATTERN YET
8169  035000  000770                          BR      30$             ;BR TO REPEAT THE PATTERN
8170  035002  005726                  44$:    TST     (SP)+           ;POP THE STACK
8171  035004  104410                          RESREG                  ;RESTORE R0-R5
8172  035006  000207                          RTS     PC              ;RETURN
8173
8174
8175
8176                                  ;;*****************************************************************
8177                                  .SBTTL  CMPBJF - SOFTWARE COMPARE DATA
8178                                  ;*THIS SUBROUTINE PERFORMS A SOFTWARE COMPARE OF THE R/W DATA BUFFER (POINTED
8179                                  ;*TO BY PMA AND PMA+2) TO THE APPROPRIATE REPEATING DATA PATTERN. IF
8180                                  ;*PARAMETER PATRN = 0 ON ENTRY, THE DATA IS COMPRISED OF ALL THE PATTERNS
8181                                  ;*00-15 (QUICK VERIFY DEFAULT DATA TEST).
8182                                  ;*IF PATRN IS NOT 0, THE NO. OF THE BIT SET IN R1 ON ENTRY IDENTIFIES THE
8183                                  ;*REPEATING DATA PATTERN TO COMPARE AGAINST.
8184                                  ;;*****************************************************************
8185  035010  104407                  CMPBUF: SAVREG                  ;SAVE R0-R5
8186  035012  112737  000040  062051          MOVB    #40,DH701+38.   ;RESTORE ERROR MSG PARAMS
8187  035020  012737  000007  063412          MOV     #7,DF25+2
8188  035026  013702  005572                  MOV     WDSXFR,R2       ;SET NO. OF WORDS
8189  035032  005037  005542                  CLR     SCRACH          ;CLEAR COMPARE ERROR COUNT
8190  035036  004737  041650                  JSR     PC,REPSUP       ;STORE PREV CMND FOR POSS. PRINT
8191  035042  005737  005544                  TST     PATRN           ;SEE IF THIS IS QUICK VERIFY DEFAULT DATA TEST
8192  035046  001007                          BNE     3$              ;BR IF NOT
8193  035050  012700  006216                  MOV     #PAT00,R0       ;GET DATA PATTERN STARTING ADDR
8194  035054  012746  000400                  MOV     #256.,-(SP)     ;SET PATTERN WORD COUNT
```

```
8195  035060  012701  063520              MOV     #RWBUF,R1       ;SET BUFFER ADDRESS
8196  035064  000466                      BR      30$             ;PROCEED
8197  035066  005000              3$:     CLR     R0              ;INIT PATTERN NO.
8198  035070  032701  000001      4$:     BIT     #BIT0,R1        ;SEE IF THIS BIT IS SET
8199  035074  001003                      BNE     6$              ;BR IF THIS BIT IS SET
8200  035076  005200                      INC     R0              ;INCREMENT PATTERN NUMBER
8201  035100  006201                      ASR     R1              ;SHIFT TO EXAMINE NEXT BIT
8202  035102  000772                      BR      4$              ;BR TO CHECK NEXT BIT
8203  035104  006300              6$:     ASL     R0              ;MULTIPLY PATTERN NO. BY 32(DEC)
8204  035106  006300                      ASL     R0
8205  035110  006300                      ASL     R0
8206  035112  006300                      ASL     R0
8207  035114  006300                      ASL     R0
8208  035116  062700  006216              ADD     #PAT00,R0       ;GET ADDRESS OF DESIRED PATTERN
8209  035122  012746  000020              MOV     #16.,-(SP)      ;PATTERN WORD COUNT
8210  035126  013701  005602              MOV     PMA,R1          ;SET BUFFER ADDRESS
8211  035132  005737  055662              TST     $KT11           ;SEE IF MEM MGT PRESENT
8212  035136  100041                      BPL     30$             ;BR IF NOT PRESENT
8213                          ;COMPARE LOOP FOR MEM MGT STARTS HERE
8214  035140  013737  003206  172354      MOV     SAVPAR,@#KIPAR6 ;SET UP WORKING PAR
8215  035146  052737  000001  177572      BIS     #BIT0,@#SR0     ;TURN ON MEM MGT
8216  035154  042701  160000              BIC     #160000,R1      ;FORCE RELOCATION THRU KIPAR6
8217  035160  052701  140000              BIS     #140000,R1
8218  035164  010003              22$:    MOV     R0,R3           ;GET A COPY OF PATTERN ADDRESS
8219  035166  011604                      MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8220  035170  022321              24$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
8221  035172  001404                      BEQ     25$             ;BR IF NO ERROR
8222  035174  013737  172354  001216      MOV     @#KIPAR6,$REG16 ;SET PAR FOR PRINTOUT
8223  035202  000432                      BR      35$             ;GO HANDLE ERROR
8224  035204  032701  020000      25$:    BIT     #BIT13,R1       ;SEE IF OVERFLOW TO NEXT PAGE
8225  035210  001405                      BEQ     26$             ;BR IF NO OVERFLOW
8226  035212  062737  000200  172354      ADD     #200,@#KIPAR6   ;INCR PAR BY 4K FOR NEW PAGE
8227  035220  042701  020000              BIC     #BIT13,R1       ;SET PAGE = 6 AGAIN
8228  035224  005302              26$:    DEC     R2              ;DECREMENT WORD COUNTER
8229  035226  001002                      BNE     27$             ;BR IF NOT ALL DONE YET
8230  035230  000137  035674              JMP     54$             ;ALL DONE - GET OUT
8231  035234  005304              27$:    DEC     R4              ;DECREMENT PATTERN WORD COUNT
8232  035236  001354                      BNE     24$             ;BR IF NOT DONE WITH PATTERN YET
8233  035240  000751                      BR      22$             ;BR TO REPEAT THE PATTERN
8234                          ;COMPARE LOOP FOR NO MEM MGT STARTS HERE
8235  035242  010003              30$:    MOV     R0,R3           ;GET COPY OF PATTERN ADDRESS
8236  035244  011604                      MOV     (SP),R4         ;INIT PATTERN WORD COUNT
8237  035246  022321              34$:    CMP     (R3)+,(R1)+     ;COMPARE DATA WORD TO PATTERN WORD
8238  035250  001002                      BNE     31$             ;BR IF COMPARE ERROR
8239  035252  000137  035654              JMP     40$             ;JUMP IF DATA COMPARES OK
8240  035256  105037  062051      31$:    CLRB    DH701+38.       ;ADJUST DATA HEADER FOR MSG
8241  035262  012737  000005  063412      MOV     #5,DF25+2       ;ADJUST ERROR DATA WORD COUNT
8242                          ;COMMON COMPARE ERROR HANDLER
8243  035270  010237  001202      35$:    MOV     R2,$REG10       ;GET WORD NO.
8244  035274  163737  005572  001202      SUB     WDSXFR,$REG10
8245  035302  013737  001202  005600      MOV     $REG10,LASTWC   ;GET 2'S COMP OF WORD NO.
8246  035310  004737  034404              JSR     PC,FINADR       ;COMPUTE ACTUAL PACK ADRS
8247  035314  104407                      SAVREG                  ;SAVE R0-R5
8248  035316  013700  005574              MOV     FINCYL,R0       ;GET CYL
8249  035322  113701  005576              MOVB    FINTRK,R1       ;GET TRACK
8250  035326  113702  005577              MOVB    FINSEC,R2       ;GET SECTOR
```

```
8251  035332  004737  040146              JSR     PC,BDSRCK       ;SEE IF THIS SECTOR LISTED BAD
8252  035336  104410                      RESREG                  ;RESTORE R0-R5
8253  035340  032737  001000  005504      BIT     #BADSEC,RECODE
8254  035346  001132                      BNE     50$             ;BR IF LISTED- DON'T REPORT ERROR
8255  035350  005737  005542              TST     SCRACH          ;CHECK THE ERROR COUNT
8256  035354  001024                      BNE     36$             ;BR IF THIS IS NOT FIRST ERROR
8257  035356  105737  003144              TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
8258  035362  001411                      BEQ     46$             ;BR IF NOT
8259  035364  013737  005270  001174      MOV     CRMPH0,$REG5    ;GET CURRENT MAP REG 0
8260  035372  013737  005266  001176      MOV     CRMPL0,$REG6
8261  035400  104116                      ERROR   116             ;DATA MISCOMPARE (11/70)
8262  035402  104117                      ERROR   117
8263  035404  000401                      BR      48$
8264  035406  104034              46$:    ERROR   34              ;TYPE HEADING FOR ERROR MSG
8265  035410  012737  177777  001174  48$:MOV     #-1,$REG5       ;INIT CYL NO.
8266  035416  005037  001176              CLR     $REG6
8267  035422  005037  001200              CLR     $REG7
8268  035426  005237  005542      36$:    INC     SCRACH          ;INCREMENT THE ERROR COUNT
8269  035432  032777  000001  143500      BIT     #BIT0,@SWR      ;SEE IF ALL ERRORS SHOULD BE REPORTED
8270  035440  001004                      BNE     38$             ;BR TO REPORT ALL ERRORS
8271  035442  022737  000012  005542      CMP     #10.,SCRACH     ;SEE IF 10(DFC) ERRORS YET
8272  035450  002511                      BLT     54$             ;BR IF ERROR LIMIT EXCEEDED
8273  035452  023737  001174  005574  38$:CMP     $REG5,FINCYL    ;SEE IF DIFFERENT CYL
8274  035460  001010                      BNE     42$             ;BR IF YES
8275  035462  123737  001176  005576      CMPB    $REG6,FINTRK    ;SEE IF DIFFERENT TRACK
8276  035470  001004                      BNE     42$             ;BR IF YES
8277  035472  123737  001200  005577      CMPB    $REG7,FINSEC    ;SEE IF DIFFERENT SECTOR
8278  035500  001412                      BEQ     44$             ;BR IF SAME PACK ADDRESS
8279  035502  013737  005574  001174  42$:MOV     FINCYL,$REG5    ;SET NEW PACK ADRS FOR PRINTOUT
8280  035510  113737  005576  001176      MOVB    FINTRK,$REG6
8281  035516  113737  005577  001200      MOVB    FINSEC,$REG7
8282  035524  104115                      ERROR   115             ;TYPE NEW PACK ADDRESS
8283  035526  005437  001202          44$:NEG     $REG10          ;GET WORD NO.
8284  035532  016337  177776  001204      MOV     -2(R3),$REG11   ;GET GOOD DATA
8285  035540  016137  177776  001206      MOV     -2(R1),$REG12   ;GET BAD DATA
8286  035546  013737  001202  001212      MOV     $REG10,$REG14   ;COMPUTE PHYSICAL ADDRESS
8287  035554  005037  001210              CLR     $REG13
8288  035560  006137  001212              ROL     $REG14
8289  035564  006137  001210              ROL     $REG13
8290  035570  063737  005602  001212      ADD     PMA,$REG14
8291  035576  005537  001210              ADC     $REG13
8292  035602  063737  005604  001210      ADD     PMA+2,$REG13
8293  035610  010137  001214              MOV     R1,$REG15       ;GET VIRT. ADRS FOR PRINTOUT
8294  035614  162737  000002  001214      SUB     #2,$REG15
8295  035622  104063                      ERROR   63              ;TYPE GOOD AND BAD DATA, MEM. ADRS.
8296  035624  032777  000100  143306      BIT     #BIT6,@SWR      ;SEE IF JUST 1 ERROR SHOULD BE REPORTED
8297  035632  001020                      BNE     54$             ;BR IF JUST 1 ERROR SHOULD BE REPORTED
8298  035634  005737  005544          50$:TST     PATRN           ;SEE IF DEFAULT DATA TEST
8299  035640  001405                      BEQ     40$             ;BR IF YES
8300  035642  005737  055662              TST     $KT11           ;SEE IF MEM MGT PRESENT
8301  035646  100002                      BPL     40$             ;BR IF NOT PRESENT
8302  035650  000137  035204              JMP     25$             ;GO HANDLE MEM MGT
8303  035654  005302                  40$:DEC     R2              ;DECREMENT WORD COUNTER
8304  035656  001406                      BEQ     54$             ;BR IF ALL DONE
8305  035660  005304                      DEC     R4              ;DECREMENT PATTERN WORD COUNT
8306  035662  001002                      BNE     53$             ;BR IF NOT DONE WITH PATTERN YET
```

```
 8307  035664  000137  035242                 JMP     30$             ;JUMP TO REPEAT THE PATTERN
 8308  035670  000137  035246          53$:   JMP     34$
 8309  035674  005737  055662          54$:   TST     $KT11           ;SEE IF MEM MGT PRESENT
 8310  035700  100003                         BPL     56$             ;BR IF NOT PRESENT
 8311  035702  042737  000001  177572         BIC     #BIT0,@#SR0     ;DISABLE MEM MGT
 8312  035710  005726                 56$:    TST     (SP)+           ;POP THE STACK
 8313  035712  104410                         RESREG                  ;RESTORE R0-R5
 8314  035714  000207                         RTS     PC              ;RETURN
 8315
 8316
 8317
 8318                                 ;;*****************************************************************
 8319                                 ;* CHKLIM - CHECK CURRENT DATA TRANSFER LIMITS
 8320                                 ;*THIS SUBROUTINE DETERMINES IF THE NEXT DATA TRANSFER SHOULD BE
 8321                                 ;*ALLOWED WITH THE CURRENT PACK ADDRESS AND WORD COUNT. IF THE
 8322                                 ;*TRANSFER WOULD CAUSE OVERFLOW BEYOND CYL 632, TRACK 1, THE
 8323                                 ;*WORD COUNT IN P.WC(R5) MUST BE SCALED DOWN TO A VALID NUMBER.
 8324                                 ;*IF NO POSSIBLE TRANSFER CAN BE ALLOWED, RETURN IS MADE TO
 8325                                 ;*ADDRESS FOLLOWING THE CALL TO CHKLIM.
 8326                                 ;* CYL 1456 USED FOR RK07
 8327                                 ;*  CALL -     JSR       PC,CHKLIM
 8328                                 ;*             <"NO TRANSFER" RETURN ADDRESS>
 8329                                 ;;*****************************************************************
 8330  035716                         CHKLIM:                         ;SAVE R0-R5
 8331  035716  026537  000002  017702         CMP     P.CYLN(R5),LCM5 ;SEE IF SHOULD CHECK ADDR. LIMITS YET
 8332  035724  002551                         BLT     34$             ;BR TO RETURN IF NOT
 8333  035726  104407                         SAVREG                  ;SAVE R0-R5
 8334  035730  005037  003176                 CLR     SUMLO1
 8335  035734  005037  003200                 CLR     SUMHI1
 8336  035740  023765  017706  000002         CMP     LSTCYL,P.CYLN(R5)       ;SEE IF ON LAST CYL
 8337  035746  001004                         BNE     8$              ;BR IF NOT
 8338  035750  122765  000002  000005         CMPB    #2,P.TRCK(R5)   ;SEE IF ON TRACK 2, CYL 632/1456
 8339  035756  001537                         BEQ     36$             ;BR IF CAN'T DO TRANSFER
 8340  035760  105737  003125          8$:    TSTB    FORMAT          ;DETERMINE THE FORMAT
 8341  035764  001411                         BEQ     10$             ;BR IF 22(DEC) SECTORS
 8342                                 ;STORE CONSTANTS FOR 20(DEC) SECTORS
 8343  035766  012746  024000                 MOV     #10240.,-(SP)   ;NO. OF WORDS ON 2 TRACKS
 8344  035772  012746  036000                 MOV     #15360.,-(SP)   ;NO. OF WORDS PER CYL
 8345  035776  012746  012000                 MOV     #5120.,-(SP)    ;NO. OF WORDS PER TRACK
 8346  036002  012746  000024                 MOV     #20.,-(SP)      ;NO. OF SECTORS
 8347  036006  000410                         BR      12$
 8348                                 ;STORE CONSTANTS FOR 22(DEC) SECTORS
 8349  036010  012746  026000          10$:   MOV     #11264.,-(SP)   ;NO. OF WORDS ON 2 TRACKS
 8350  036014  012746  041000                 MOV     #16896.,-(SP)   ;NO. OF WORDS PER CYL
 8351  036020  012746  013000                 MOV     #5632.,-(SP)    ;NO. OF WORDS PER TRACK
 8352  036024  012746  000026                 MOV     #22.,-(SP)      ;NO. OF SECTORS
 8353                                 ;COMPUTE NO. OF WORDS LEFT ON THIS TRACK
 8354  036030  012603                 12$:    MOV     (SP)+,R3        ;GET NO. OF SECTORS
 8355  036032  116502  000004                 MOVB    P.SECT(R5),R2   ;GET CURRENT SECTOR NO.
 8356  036036  160203                         SUB     R2,R3           ;NO. OF SECTORS LEFT
 8357  036040  005002                         CLR     R2              ;GET NO. OF WDS IN THESE SECTORS
 8358  036042  012705  000400                 MOV     #256.,R5
 8359  036046  005004                         CLR     R4
 8360  036050  004737  053150                 JSR     PC,M.DPIM
 8361  036054  010337  003176                 MOV     R3,SUMLO1       ;STORE THIS NO.
 8362  036060  016605  000012                 MOV     12(SP),R5       ;RESTORE PARAM BLK ADDR
```

```
8363                                    ;COMPUTE NO. OF WORDS LEFT ON THIS CYLINDER
8364  036064  012703  000002                    MOV     #2,R3           ;TRACK LIMIT = 2
8365  036070  023765  017706  000002            CMP     LSTCYL,P.CYLN(R5) ;SEE IF ON CYL 632/1456
8366  036076  001001                            BNE     14$             ;BR IF NOT
8367  036100  005303                            DEC     R3              ;DECREMENT TRACK LIMIT TO 1 FOR CYL 632/1456
8368  036102  116502  000005            14$:    MOVB    P.TRCK(R5),R2   ;GET CURRENT TRACK NO.
8369  036106  160203                            SUB     R2,R3           ;GET NO. OF TRACKS LEFT
8370  036110  005002                            CLR     R2              ;GET NO. OF WORDS IN THESE TRACKS
8371  036112  012605                            MOV     (SP)+,R5        ;NO. OF WDS PER TRACK
8372  036114  004737  053150                    JSR     PC,M.DPIM
8373  036120  060337  003176                    ADD     R3,SUMLO1       ;ADD WORDS TO TOTAL
8374  036124  016605  000010                    MOV     10(SP),R5       ;RESTORE PARAM BLK ADR
8375                                    ;COMPUTE NO. OF WORDS ON WHOLE CYLINDERS REMAINING
8376  036130  013703  017704                    MOV     LCM1,R3         ;CYL LIMIT =631/1455
8377  036134  166503  000002                    SUB     P.CYLN(R5),R3   ;GET NO. OF WHOLE CYLS LEFT
8378  036140  100001                            BPL     16$
8379  036142  005003                            CLR     R3
8380  036144  005002            16$:    CLR     R2
8381  036146  012605                            MOV     (SP)+,R5        ;GET NO. OF WDS PER CYL
8382  036150  004737  053150                    JSR     PC,M.DPIM       ;COMPUTE NO. OF WDS IN THESE WHOLE CYLS
8383  036154  063703  003176                    ADD     SUMLO1,R3       ;ADD THESE WDS TO TOTAL
8384  036160  005502                            ADC     R2
8385  036162  063702  003200                    ADD     SUMHI1,R2
8386  036166  016605  000006                    MOV     6(SP),R5        ;RESTORE PARAM BLK ADDR
8387                                    ;ADD THE NO. OF WDS ON TRACKS 0,1, CYL 632/1456
8388  036172  023765  017706  000002            CMP     LSTCYL,P.CYLN(R5)       ;SEE IF CYL = 632/1456
8389  036200  001002                            BNE     18$             ;BR IF NOT 632/1456
8390  036202  005726                            TST     (SP)+           ;POP STACK
8391  036204  000402                            BR      20$
8392  036206  062603            18$:    ADD     (SP)+,R3        ;ADD WDS ON TRKS 0,1, CYL 632/1456
8393  036210  005502                            ADC     R2
8394                                    ;GET DESIRED WORD COUNT
8395  036212  005000            20$:    CLR     R0              ;WORD COUNT HI BITS
8396  036214  016501  000012                    MOV     P.WC(R5),R1     ;GET THE DESIRED WORD COUNT
8397  036220  005401                            NEG     R1
8398  036222  001001                            BNE     22$             ;BR IF WORD COUNT NOT 65,536(DEC)
8399  036224  005200                            INC     R0              ;SET HI WORD COUNT = 1
8400                                    ;SUBTRACT WORD COUNT FROM NO. OF WORDS LEFT TO DETERMINE IF OVERFLOW.
8401                                    ;   NUMBER OF WORDS LEFT IS IN R2-R3, WORD COUNT IS IN R0-R1.
8402  036226  160103            22$:    SUB     R1,R3           ;LO PARTS
8403  036230  005602                            SBC     R2
8404  036232  160002                            SUB     R0,R2           ;HI PARTS
8405  036234  100004                            BPL     30$             ;BR IF WORD COUNT NOT > NO. OF WDS LEFT
8406                                    ;SCALE DOWN WORD COUNT TO AVOID OVERFLOW (SUBTRACT THE EXCESS)
8407  036236  060301                            ADD     R3,R1           ;GET NEW WORD COUNT IN R1
8408  036240  005401                            NEG     R1
8409  036242  010165  000012                    MOV     R1,P.WC(R5)     ;SET NEW WORD COUNT IN PARAM BLOCK
8410                                    ;RETURN
8411  036246  104410            30$:    RESREG                  ;RESTORE R0-R5
8412  036250  062716  000002    34$:    ADD     #2,(SP)         ;FIX NORMAL RETURN PC
8413  036254  000207                            RTS     PC              ;EXIT
8414  036256  017616  000000    36$:    MOV     @(SP),(SP)      ;FIX "NO TRANSFER" RETURN PC
8415  036262  104410                            RESREG                  ;RESTORE R0-R5
8416  036264  000207                            RTS     PC              ;RETURN
8417
8418
```

```
8419
8420                                    ;;*****************************************************************
8421                                    ;* CTLOUT - THIS SUBROUTINE CHECKS FOR (^C) OR (^Z) TTY INPUT.
8422                                    ;*IF (^C) WAS TYPED, THE SUBROUTINE RETURNS TO DRVTST. IF (^Z)
8423                                    ;*WAS TYPED, THE SUBROUTINE RETURNS TO INPUTP. IN EITHER CASE,
8424                                    ;*A RESET INSTRUCTION IS EXECUTED, AND THE STACK IS RE-INIT-
8425                                    ;*IALIZED. IF THERE IS NO INPUT, OR INPUT OTHER THAN (^C) OR (^Z),
8426                                    ;*NO ACTION IS TAKEN.
8427                                    ;*  CALL -   JSR     PC,CTLOUT
8428                                    ;*    OR -   CKEXIT
8429                                    ;;*****************************************************************
8430  036266  122737  000001  003120  CTLOUT: CMPB    #1,TSTING       ;SEE IF CURRENTLY RUNNING TESTS
8431  036274  001077                          BNE     10$             ;BR IF NOT RUNNING TESTS
8432  036276  005737  005532                  TST     INTCHR          ;SEE IF ANY TTY INPUT
8433  036302  001474                          BEQ     10$             ;BR IF NO INPUT
8434  036304  105737  003116                  TSTB    MDFLAG          ;SEE IF DEFAULT MODE RUN
8435  036310  001446                          BEQ     12$             ;BR IF YES
8436  036312  122737  000003  005532          CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
8437  036320  001033                          BNE     4$              ;BR IF NOT (^C)
8438  036322  012700  013660                  MOV     #DRVTST,R0      ;SET RETURN ADDR = DRVTST
8439  036326  105737  003134          2$:     TSTB    XOVLAD          ;SEE IF XXDP CURRENTLY OVERLAID
8440  036332  001402                          BEQ     6$              ;BR IF NOT
8441  036334  004737  027210                  JSR     PC,GETXDP       ;RESTORE SAVED XXDP, IF NECESSARY
8442  036340  000005                  6$:     RESET                   ;RESET ALL DEVICES
8443  036342  005037  005532                  CLR     INTCHR          ;CLEAR TTY CHAR BUFFER WORD
8444  036346  005037  001304                  CLR     $TIMES          ;CLEAR THE ITER. COUNT
8445  036352  105037  003135                  CLRB    XDPSVD          ;CLEAR THE XXDP SAVED FLAG
8446  036356  012737  042130  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADDRESS
8447  036364  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK
8448  036370  112765  000113  000001          MOVB    #RECAL,P.CMND(R5) ;SET RECAL COMMAND
8449  036376  004737  040252                  JSR     PC,DRVCAL       ;DO CLEANUP RECALIBRATE
8450  036402  005037  001102                  CLR     $TSTNM          ;CLEAR THE TEST NO.
8451  036406  000110                          JMP     @R0             ;EXIT FROM TESTS
8452  036410  122737  000032  005532  4$:     CMPB    #032,INTCHR     ;SEE IF (^Z) TYPED
8453  036416  001016                          BNE     7$              ;BR IF NOT (^Z)
8454  036420  012700  015266                  MOV     #INPUTP,R0      ;SET RETURN ADDR = INPUTP
8455  036424  000740                          BR      2$              ;TAKE EXIT
8456  036426  122737  000003  005532  12$:    CMPB    #003,INTCHR     ;SEE IF (^C) TYPED
8457  036434  001007                          BNE     7$              ;BR IF NOT
8458  036436  104401  010062                  TYPE    ,HLTRQD         ;TYPE ''HALT REQUESTED''
8459  036442  104401  010032                  TYPE    ,CNTRDY         ;TYPE ''PRESS CONT WHEN RDY''
8460  036446  012700  044100                  MOV     #HLTPRG,R0      ;SET HALT ADDRESS
8461  036452  000725                          BR      2$              ;TAKE EXIT
8462  036454  122737  000007  005532  7$:     CMPB    #007,INTCHR     ;SEE IF (^G) TYPED
8463  036462  001002                          BNE     8$              ;BR IF NOT (^G)
8464  036464  004737  026316                  JSR     PC,GTSWRG       ;OPEN SOFTWARE SWR FOR MODIFICATION
8465  036470  004737  025766          8$:     JSR     PC,PREPKB       ;ENABLE KBD INPUT AGAIN
8466  036474  000207                  10$:    RTS     PC              ;RETURN
8467
8468
8469
8470                                    ;;*****************************************************************
8471                                    ;*WRTSEC - WRITE A BLOCK OF 400(OCT) WORDS ONTO DISK AND PERFORM
8472                                    ;*A WRITE CHECK. PARAMETER BLOCK MUST BE PRE-LOADED WITH PACK
8473                                    ;*ADDRESS. DATA WORD IS PASSED IN R3 ON ENTRY. RWBUF IS THE BUFFER
8474                                    ;*USED.
```

```
                                ;;*****************************************************************
036476  104407                  WRTSEC: SAVREG                  ;SAVE R0-R5
                                ;LOAD THE R/W BUFFER WITH DATA
036500  012700  063520                  MOV     #RWBUF,R0       ;BUFFER ADDRESS
036504  012701  000400                  MOV     #400,R1         ;400(OCT) WORDS
036510  010320                  4$:     MOV     R3,(R0)+        ;LOAD A BUFFER WORD
036512  005301                          DEC     R1              ;DECR COUNTER
036514  001375                          BNE     4$              ;BR IF NOT DONE YET
                                ;SET UP PARAMETERS
036516  012765  063520  000010          MOV     #RWBUF,P.BALO(R5) ;SET BUS ADDRESS
036524  012765  177400  000012          MOV     #-400,P.WC(R5)  ;SET WORD COUNT
                                ;WRITE THE DATA
036532  112765  000123  000001          MOVB    #WRDATA,P.CMND(R5) ;SET WRITE COMMAND
036540  004737  040252                  JSR     PC,DRVCAL       ;WRITE THE DATA
                                ;PERFORM WRITE CHECK
036544  112765  000131  000001  6$:     MOVB    #WRTCHK,P.CMND(R5) ;SET WRITE CHECK COMMAND
036552  004737  040252                  JSR     PC,DRVCAL       ;PERFORM WRITE CHECK
036556  104410                          RESREG                  ;RESTORE R0-R5
036560  000207                          RTS     PC              ;RETURN


                                ;;*****************************************************************
                                ;*INCRMA - INCREMENT MA BY WORD COUNT IN P.WC(R5).
                                ;;*****************************************************************
036562  104407                  INCRMA: SAVREG                  ;SAVE R0-R5
036564  005001                          CLR     R1
036566  016500  000012                  MOV     P.WC(R5),R0     ;GET WORD COUNT
036572  001002                          BNE     4$              ;BR IF NOT 65,536(DEC)
036574  005201                          INC     R1              ;SET HI BIT
036576  000401                          BR      6$              ;CONTINUE
036600  005400                  4$:     NEG     R0              ;GET TRUE WORD COUNT
036602  000241                  6$:     CLC                     ;DOUBLE THE WORD COUNT TO GET BYTES
036604  006100                          ROL     R0
036606  006101                          ROL     R1
036610  060037  005762                  ADD     R0,MA           ;ADD IT TO COMPUTE NEW MA
036614  005537  005764                  ADC     MA+2
036620  060137  005764                  ADD     R1,MA+2
036624  104410                          RESREG                  ;RESTORE R0-R5
036626  000207                          RTS     PC              ;RETURN
                                


                                ;;*****************************************************************
                                ;*REDBSF - READ FACTORY BSF INTO BSFACT, AND READ SOFTWARE BSF
                                ;*INTO BSSOFT.
                                ;;*****************************************************************
036630                          REDBSF:
036630  004737  027450                  JSR     PC,INITSS       ;INIT THE S.S.
036634  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5)      ;SET FOR 22 SECTOR FORMAT
036642  112765  000121  000001          MOVB    #RDDATA,P.CMND(R5) ;SET READ DATA COMMAND
036650  013765  017706  000002          MOV     LSTCYL,P.CYLN(R5)       ;SET CYL = 632/1456
036656  112765  000002  000005          MOVB    #2,P.TRCK(R5)   ;SET TRACK = 2
036664  012703  000012                  MOV     #10.,R3         ;SET FACTORY BSF SECTOR LIMIT
036670  105737  003125                  TSTB    FORMAT
036674  001403                          BEQ     6$              ;BR IF 22 SECTORS
```

```
 8531  036676  105265  000004                  INCB    P.SECT(R5)      ;SET STARTING FACTORY BSF SECTOR NO.
 8532  036702  005203                          INC     R3
 8533  036704  012765  177400  000012  6$:     MOV     #-400,P.WC(R5)  ;SET WORD COUNT FOR 1 SECTOR
 8534  036712  012765  004232  000010          MOV     #BSFACT,P.BALO(R5) ;SET BA FOR FACTORY BSF
 8535  036720  012737  037074  003046  8$:     MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
 8536  036726  105037  003141                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
 8537  036732  004737  040252                  JSR     PC,DRVCAL       ;READ THE FACTORY BSF
 8538  036736  105737  003141                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
 8539  036742  001413                          BEQ     12$             ;BR IF NOT
 8540  036744  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
 8541  036752  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
 8542  036756  001360                          BNE     8$              ;BR IF NOT YET
 8543  036760  004737  041650          10$:    JSR     PC,REPSUP       ;GATHER STATUS FOR PRINTOUT
 8544  036764  104120                          ERROR   120             ;ABORTING- BAD BSF READ
 8545  036766  000137  044100                  JMP     HLTPRG          ;HALT- CAN'T PROCEED
 8546  036772  012765  003232  000010  12$:    MOV     #BSSOFT,P.BALO(R5) ;SET BA FOR SOFTWARE RECORD
 8547  037000  110365  000004                  MOVB    R3,P.SECT(R5)   ;SET STARTING SECTOR NO.
 8548  037004  012703  000026                  MOV     #22.,R3         ;SET 22 SECTOR LIMIT
 8549  037010  105737  003125                  TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
 8550  037014  001402                          BEQ     14$             ;BR IF YES
 8551  037016  012703  000023                  MOV     #19.,R3         ;SET 20 SECTOR LIMIT
 8552  037022  012737  037074  003046  14$:    MOV     #BDSCHD,A.ABNL ;SET ERROR HANDLER FOR BSF READ
 8553  037030  105037  003141                  CLRB    WCEFLG          ;INIT THE ERROR FLAG
 8554  037034  004737  040252                  JSR     PC,DRVCAL       ;READ THE SOFTWARE BSF
 8555  037040  105737  003141                  TSTB    WCEFLG          ;SEE IF ANY ERRORS
 8556  037044  001407                          BEQ     16$             ;BR IF NOT
 8557  037046  062765  000002  000004          ADD     #2,P.SECT(R5)   ;CHECK NEXT SECTOR
 8558  037054  126503  000004                  CMPB    P.SECT(R5),R3   ;SEE IF LIMIT EXCEEDED YET
 8559  037060  003760                          BLE     14$             ;BR IF NOT YET
 8560  037062  000736                          BR      10$             ;REPORT ERROR AND ABORT
 8561  037064  012737  042130  003046  16$:    MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR HANDLER ADRS
 8562  037072  000207                          RTS     PC              ;RETURN
 8563
 8564                                  ;*THIS IS THE ABNORMAL DRIVER RETURN WHICH IS USED TO HANDLE
 8565                                  ;*POSSIBLE ERRORS IN READING BAD SECTORS.
 8566  037074  032765  130600  000034  BDSCHD: BIT     #BSE!HVRC!DTE!OPI!DCK,P.ER(R5) ;SEE IF ANY READ ERRORS
 8567  037102  001002                          BNE     4$              ;BR IF A READ ERROR OCCURRED
 8568  037104  000137  042130                  JMP     ERRHDL          ;GO HANDLE OTHER ERROR
 8569  037110  105237  003141          4$:     INCB    WCEFLG          ;SET ERROR FLAG
 8570  037114  000137  044306                  JMP     RETNML          ;TAKE NORMAL DRIVER RETURN
 8571
 8572
 8573
 8574                                  ;;*****************************************************************
 8575                                  ;*TYPBSF - TYPE FACTORY AND SOFTWARE BAD SECTOR FILES (BSF'S)
 8576                                  ;*WITH THE CYLINDER, TRACK, AND SECTOR NO. OF EACH BAD SECTOR
 8577                                  ;*LISTED SEPARATELY, IN A TABLE OF OCTAL VALUES, FOR EACH OF THE
 8578                                  ;*2 BSF'S.
 8579                                  ;;*****************************************************************
 8580  037120  104407                  TYPBSF: SAVREG                  ;SAVE R0-R5
 8581  037122  005004                          CLR     R4              ;INIT BSF FILE INDICATOR
 8582  037124  012700  004242                  MOV     #BSFACT+10,R0   ;ADRS OF DATA IN FACTORY BSF
 8583  037130  104401  011364                  TYPE    ,FACTBS         ;TYPE "FACTORY"
 8584  037134  104401  011411                  TYPE    ,BDSECT         ;TYPE "BAD SECTORS"
 8585  037140  104401  061616          3$:     TYPE    ,DH6041         ;TYPE   "CYLNDR  TRACK   SECTOR"
 8586  037144  104401  001315                  TYPE    ,$CRLF
```

```
8587  037150  005001                    CLR     R1              ;INIT LIMIT COUNTER
8588  037152  005710              4$:   TST     (R0)            ;SEE IF A SECTOR LISTED HERE
8589  037154  100007                    BPL     8$              ;BR IF YES
8590  037156  005701                    TST     R1              ;SEE IF FILE IS EMPTY
8591  037160  001032                    BNE     14$             ;BR IF NOT EMPTY
8592  037162  104401  011431            TYPE    ,NOFALS         ;TYPE "NONE"
8593  037166  104401  001315            TYPE    ,$CRLF          ;TYPE <CR>,<LF>
8594  037172  000425                    BR      14$
8595  037174  012046              8$:   MOV     (R0)+,-(SP)     ;GET A CYL NO.
8596  037176  104402                    TYPOC                   ;TYPE IT
8597  037200  104401  012420            TYPE    ,SPACE2         ;TYPE SEPARATORS
8598  037204  011003                    MOV     (R0),R3         ;GET TRACK AND SECTOR
8599  037206  105003                    CLRB    R3
8600  037210  000303                    SWAB    R3              ;GET THE TRACK NO.
8601  037212  010346                    MOV     R3,-(SP)
8602  037214  104402                    TYPOC                   ;TYPE IT
8603  037216  104401  012420            TYPE    ,SPACE2         ;TYPE SEPARATORS
8604  037222  111003                    MOVB    (R0),R3         ;GET SECTOR NO.
8605  037224  010346                    MOV     R3,-(SP)
8606  037226  104402                    TYPOC                   ;TYPE IT
8607  037230  104401  001315            TYPE    ,$CRLF
8608  037234  005720                    TST     (R0)+           ;INCR THE POINTER
8609  037236  005201                    INC     R1              ;INCR THE COUNTER
8610  037240  020127  000176            CMP     R1,#126.        ;SEE IF LIMIT REACHED IN THIS FILE
8611  037244  002742                    BLT     4$              ;BR IF NOT YET
8612  037246  005704              14$:  TST     R4              ;SEE IF BOTH FILES TYPED YET
8613  037250  001006                    BNE     18$             ;BR IF YES
8614  037252  005204                    INC     R4              ;SET INDICATOR FOR SOFT. FILE
8615  037254  012700  003242            MOV     #BSSOFT+10,R0   ;ADRS OF DATA IN SOFTWARE BSF
8616  037260  104401  011377            TYPE    ,SOFTBS         ;TYPE "SOFTWARE BAD SECTORS :"
8617  037264  000725                    BR      3$              ;GO TYPE SOFT. BSF
8618  037266  104410              18$:  RESREG                  ;RESTORE R0-R5
8619  037270  000207                    RTS     PC              ;RETURN
8620
8621
8622
8623                                    ;;*****************************************************
8624                                    ;*FINMEM - COMPUTE NO. OF WORDS XFERRED ON PARTIAL XFER
8625                                    ;*(TERMINATED BY BSE ERROR), AND STORE IT IN WDSXFR.
8626                                    ;;*****************************************************
8627  037272  104407              FINMFM: SAVREG                ;SAVE R0-R5
8628  037274  016500  000002            MOV     P.CYLN(R5),R0   ;GET ORIG. CYL NO.
8629  037300  116501  000005            MOVB    P.TRCK(R5),R1   ;GET TRACK NO.
8630  037304  116502  000004            MOVB    P.SECT(R5),R2   ;SECTOR NO.
8631  037310  005003                    CLR     R3              ;INIT SECTOR COUNT TO 0
8632  037312  026500  000030      6$:   CMP     P.DCYL(R5),R0   ;COMPARE ORIG. AND CURRENT CYLS
8633  037316  001006                    BNE     8$              ;BR IF NOT EQUAL
8634  037320  126501  000027            CMPB    P.DTS+1(R5),R1  ;COMPARE TRACKS
8635  037324  001003                    BNE     8$              ;BR IF NOT EQUAL
8636  037326  126502  000026            CMPB    P.DTS(R5),R2    ;COMPARE SECTORS
8637  037332  001404                    BEQ     12$             ;BR IF ADRS ARE EQUAL
8638  037334  005203              8$:   INC     R3              ;INCR SECTOR COUNT
8639  037336  004737  037356            JSR     PC,INCRSC       ;INCR PACK ADRS BY 1 SECTOR
8640  037342  000763                    BR      6$              ;GO COMPARE ADDRESSES
8641  037344  000303              12$:  SWAB    R3              ;COMPUTE NO. OF WORDS XFERRED
8642  037346  010337  005272            MOV     R3,WDSXFR       ;STORE IT
```

```
8643  037352  104410                  RESREG                  ;RESTORE R0-R5
8644  037354  000207                  RTS     PC              ;RETURN
8645
8646
8647
8648                          ;;*****************************************************************
8649                          ;*INCRSC - INCREMENT PACK ADDRESS TO NEXT SECTOR
8650                          ;*THE CYLINDER IS PASSED AND RETURNED IN R0, TRACK IN R1, AND SECTOR
8651                          ;*IN R2.
8652                          ;;*****************************************************************
8653  037356  012704  000025  INCRSC: MOV     #21.,R4         ;SET SECTOR LIMIT
8654  037362  105737  003125          TSTB    FORMAT          ;SEE IF 22 SECTOR FORMAT
8655  037366  001402                  BEQ     4$              ;BR IF YES
8656  037370  012704  000023          MOV     #19.,R4         ;SET SECTOR LIMIT
8657  037374  005202          4$:     INC     R2              ;INCR. SECTOR NO.
8658  037376  020204                  CMP     R2,R4           ;SEE IF SECTOR LIMIT EXCEEDED
8659  037400  003407                  BLE     6$              ;BR IF NOT
8660  037402  005002                  CLR     R2              ;SET SECTOR = 0
8661  037404  005201                  INC     R1              ;INCR. TRACK NO.
8662  037406  020127  000002          CMP     R1,#2           ;SEE IF TRACK LIMIT EXCEEDED
8663  037412  003402                  BLE     6$              ;BR IF NOT
8664  037414  005001                  CLR     R1              ;SET TRACK = 0
8665  037416  005200                  INC     R0              ;INCR. CYLINDER NO.
8666  037420  000207          6$:     RTS     PC              ;RETURN
8667
8668
8669
8670                          ;;*****************************************************************
8671                          ;*MIDXFR - MID-TRANSFER PARAMETER UPDATE SUBROUTINE
8672                          ;*THIS SUBROUTINE UPDATES PMA,PMA+2,MEM MGT REGS,UNIBUS MAP,P.BALO(R5),
8673                          ;*P.BAHI(R5), AND P.WC(R5), IN PREPARATION TO RESUME A TRANSFER
8674                          ;*TERMINATED BY A BAD SECTOR ERROR (BSE).
8675                          ;;*****************************************************************
8676  037422  104407          MIDXFR: SAVREG                  ;SAVE R0-R5
8677  037424  004737  037272          JSR     PC,FINMEM       ;COMPUTE NO. OF WORDS XFERRED
8678  037430  016500  000030          MOV     P.DCYL(R5),R0   ;GET CYL NO.
8679  037434  116501  000027          MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
8680  037440  116502  000026          MOVB    P.DTS(R5),R2    ;GET SECTOR NO.
8681  037444  004737  037356          JSR     PC,INCRSC       ;INCREMENT PACK ADRS TO NEXT SECTOR
8682  037450  010065  000002          MOV     R0,P.CYLN(R5)   ;UPDATE CYLINDER
8683  037454  110165  000005          MOVB    R1,P.TRCK(R5)   ;UPDATE TRACK
8684  037460  110265  000004          MOVB    R2,P.SECT(R5)   ;UPDATE SECTOR
8685  037464  013703  005572          MOV     WDSXFR,R3       ;GET NO. OF WORDS XFERRED
8686  037470  062703  000400          ADD     #400,R3         ;SKIP BAD SECTOR
8687  037474  060365  000012          ADD     R3,P.WC(R5)     ;UPDATE P.WC(R5)
8688  037500  005004                  CLR     R4              ;GET BYTES XFERRED
8689  037502  006103                  ROL     R3
8690  037504  006104                  ROL     R4
8691  037506  060337  005602          ADD     R3,PMA          ;UPDATE PMA,PMA+2
8692  037512  005537  005604          ADC     PMA+2
8693  037516  060437  005604          ADD     R4,PMA+2
8694  037522  013765  005602  000010  MOV     PMA,P.BALO(R5)  ;UPDATE P.BALO(R5)
8695  037530  013700  005604          MOV     PMA+2,R0
8696  037534  042700  177774          BIC     #177774,R0
8697  037540  150065  000007          BISB    R0,P.BAHI(R5)   ;UPDATE P.BAHI(R5)
8698  037544  005737  055662          TST     $KT11           ;SEE IF MEM MGT
```

```
8699  037550  100002                        BPL     16$
8700  037552  004737  026532                JSR     PC,PREPAR       ;UPDATE MEM MGT AND UNIBUS MAP
8701  037556  104410                16$:    RESREG                  ;RESTORE R0-R5
8702  037560  000207                        RTS     PC              ;RETURN
8703
8704
8705
8706                                ;;*****************************************************************
8707                                ;*SVPRMS - SAVE INITIAL PARAMETERS FOR TRANSFER
8708                                ;*GTPRMS - RESTORE SAVED TRANSFER PARAMETERS
8709                                ;;*****************************************************************
8710  037562  104407                SVPRMS: SAVREG                  ;SAVE R0-R5
8711  037564  012700  002632                MOV     #PARM0+2,R0     ;ADRS OF PARAMS
8712  037570  012701  005560                MOV     #SAVPRS,R1      ;ADRS OF SAVE AREA
8713  037574  016537  000012  005572        MOV     P.WC(R5),WDSXFR  ;GET WORD COUNT
8714  037602  005437  005572                NEG     WDSXFR          ;MAKE IT POSITIVE
8715  037606  012737  063520  005602        MOV     #RWBUF,PMA      ;INIT PMA TO RWBUF
8716  037614  005037  005604                CLR     PMA+2           ;INIT PMA+2 TO 0
8717  037620  005737  005544                TST     PATRN           ;SEE IF DEFAULT DATA TEST
8718  037624  001414                        BEQ     GTO             ;BR IF YES
8719  037626  013737  005762  005602        MOV     MA,PMA          ;SET PMA=MA
8720  037634  013737  005764  005604        MOV     MA+2,PMA+2      ;SET PMA+2=MA+2
8721  037642  000405                        BR      GTO
8722  037644  104407                GTPRMS: SAVREG                  ;SAVE R0-R5
8723  037646  012700  005560                MOV     #SAVPRS,R0      ;ADRS OF SAVE AREA
8724  037652  012701  002632                MOV     #PARM0+2,R1     ;ADRS OF PARAMS
8725  037656  012702  000005        GTO:    MOV     #5,R2           ;SET FOR 5 WORDS
8726  037662  012021                6$:     MOV     (R0)+,(R1)+     ;MOVE A WORD
8727  037664  005302                        DEC     R2              ;SEE IF DONE YET
8728  037666  001375                        BNE     6$              ;BR IF NOT YET
8729  037670  104410                        RESREG                  ;RESTORE R0-R5
8730  037672  000207                        RTS     PC              ;RETURN
8731
8732
8733
8734                                ;;*****************************************************************
8735                                ;*TRNSFR - PERFORM A DATA TRANSFER, AND HANDLE BAD SECTORS, IF
8736                                ;*ENCOUNTERED. THE COMMAND MUST BE SET IN THE PARAM BLK. THIS SUB-
8737                                ;*ROUTINE IS SUBSTITUTED FOR DRVCAL WHEN BAD SECTORS MUST BE HANDLED.
8738                                ;;*****************************************************************
8739  037674  010446                TRNSFR: MOV     R4,-(SP)        ;SAVE R4 ON STACK
8740  037676  005004                        CLR     R4              ;CLEAR BSE ERROR INDICATOR
8741  037700  105037  003141                CLRB    WCEFLG          ;INIT WCE ERROR FLAG TO 0
8742  037704  004737  040252        4$:     JSR     PC,DRVCAL       ;PERFORM THE S.S. FUNCTION
8743  037710  032737  000100  005504        BIT     #WCERR,RECODE   ;SEE IF WCE ERROR OCCURRED
8744  037716  001403                        BEQ     5$              ;BR IF NOT
8745  037720  112737  000001  003141        MOVB    #1,WCEFLG       ;SET WCE ERROR FLAG
8746  037726  032737  000002  005504  5$:   BIT     #BSERR,RECODE   ;SEE IF BSE ERROR OCCURRED
8747  037734  001406                        BEQ     6$              ;BR IF NOT
8748  037736  005204                        INC     R4              ;INCR BSE ERROR INDICATOR
8749  037740  004737  037422                JSR     PC,MIDXFR       ;UPDATE PARAMS TO RESUME XFER
8750  037744  005765  000012                TST     P.WC(R5)        ;SEE IF ENTIRE XFER IS COMPLETED
8751  037750  001355                        BNE     4$              ;BR IF NOT
8752  037752  005704                6$:     TST     R4              ;SEE IF ANY BSE ERRORS OCCURRED
8753  037754  001411                        BEQ     8$              ;BR IF NOT
8754  037756  004737  037644                JSR     PC,GTPRMS       ;RESTORE ORIGINAL PARAMS OF XFER
```

```
 8755  037762  004737  037562            JSR     PC,SVPRMS       ;RESTORE WDSXFR,PMA,PMA+2
 8756  037766  005737  055662            TST     $KT11           ;SEE IF MEM MGT PRESENT
 8757  037772  100002                    BPL     8$              ;BR IF NOT
 8758  037774  004737  026532            JSR     PC,PREPAR       ;PREPARE MEM MGT AND U.M.
 8759  040000  012604            8$:     MOV     (SP)+,R4        ;RESTORE R4
 8760  040002  000207                    RTS     PC              ;RETURN
 8761
 8762
 8763
 8764                                    ;;*****************************************************************
 8765                                    .SBTTL  SEARCH BAD SECTOR TABLES ROUTINE
 8766                                    ;*THIS ROUTINE RETURNS A LIST OF BAD SECTORS FOUND IN
 8767                                    ;*THE SPECIFIC TABLE (BAD SECTOR SOFTWARE OR BAD SECTOR FACTORY)
 8768                                    ;*FOR THE SPECIFIC CYLINDER AND TRACK DESIRED. R0 AND R1 MUST
 8769                                    ;*CONTAIN THE CYLINDER AND TRACK, RESPECTIVELY.
 8770                                    ;*
 8771                                    ;*THE LIST OF BAD SECTORS IS FOUND ON THE STACK WHEN THE ROUTINE
 8772                                    ;*RETURNS TO THE CALLER. THE FIRST ENTRY ON THE STACK WILL BE THE
 8773                                    ;*NUMBER OF BAD SECTORS FOUND FOR THIS CYLINDER AND TRACK.
 8774                                    ;;*****************************************************************
 8775  040004  010237  001266    SRHTBS: MOV     R2,$TMP2
 8776  040010  010337  001270            MOV     R3,$TMP3
 8777  040014  012637  001272            MOV     (SP)+,$TMP4     ;STORE RETURN CONTENTS OF R4
 8778  040020  011402                    MOV     (R4),R2 ;GET ADDRESS OF BAD SECTOR TABLE TO SEARCH
 8779  040022  012437  001264            MOV     (R4)+,$TMP1     ;SETUP FOR LENGTH OF BAD SECTOR TABLE
 8780  040026  062737  001000  001264    ADD     #1000,$TMP1
 8781  040034  005003                    CLR     R3              ;CLEAR R3 FOR COUNT
 8782  040036  062702  000010            ADD     #10,R2          ;SET R2 FOR POINTER TO CYLINDER ENTRY
 8783  040042  005712            1$:     TST     (R2)            ;TEST IF ALL ONES
 8784  040044  100430                    BMI     5$              ;YES-DONE
 8785  040046  020012                    CMP     R0,(R2)         ;TEST IF BAD SECTOR IN PRESENT CYL
 8786  040050  001406                    BEQ     3$              ;YES-GO CHECK TRACK
 8787  040052  062702  000004            ADD     #4,R2           ;ELSE BUMP POINTER
 8788  040056  020237  001264            CMP     R2,$TMP1        ;TEST IF OUT OF TABLE
 8739  040062  002021                    BGE     5$              ;YES-EXIT
 8790  040064  000766                    BR      1$              ;LOOP
 8791  040066  005722            3$:     TST     (R2)+           ;TRACK CHECK - PUT POINTER AT TRK/SEC BYTE
 8792  040070  011237  001302            MOV     (R2),$TMP10     ;GET TRK/SEC WORD
 8793  040074  042737  174377  001302    BIC     #174377,$TMP10  ;CLEAR ALL BUT TRACK
 8794  040102  123701  001303            CMPB    $TMP10+1,R1     ;CHECK IF BAD SECTOR IN THIS TRACK
 8795  040106  001402                    BEQ     4$              ;YES - GO PUT SECTOR NUMBER ON STACK
 8796  040110  005722                    TST     (R2)+           ;ELSE BUMP POINTER TO NEXT CYL WORD
 8797  040112  000753                    BR      1$              ;GO TEST NEXT CYL
 8798  040114  005203            4$:     INC     R3              ;BUMP BAD SECTOR COUNT
 8799  040116  012246                    MOV     (R2)+,-(SP)     ;PUT TRK/SEC WORD ON STACK
 8800  040120  042716  177700            BIC     #177700,(SP)    ;CLEAR ALL BUT SECTOR NUMBER
 8801  040124  000746                    BR      1$              ;GO CHECK REST OF FILE
 8802  040126  010346            5$:     MOV     R3,-(SP)        ;EXIT - PUT NUMBER OF BAD
 8803  040130  013702  001266            MOV     $TMP2,R2        ;SECTORS ON STACK-RESTORE
 8804  040134  013703  001270            MOV     $TMP3,R3        ;REGISTERS
 8805  040140  013746  001272            MOV     $TMP4,-(SP)     ;PUT RETURN ON STACK
 8806  040144  000204                    RTS     R4              ;RETURN
 8807                                    ;;*****************************************************************
 8808                                    .SBTTL  BAD SECTOR CHECK ROUTINE
 8809                                    ;*THIS ROUTINE WILL SEARCH BOTH TABLES TO DETERMINE IF A
 8810                                    ;*SPECIFIC SECTOR IS LISTED AS BAD. IF THE SECTOR IS LISTED IN THE
```

```
8811                                    ;*TABLES THE ROUTINE SETS THE "BADSEC" FLAG AND RETURNS. IF THE SECTOR
8812                                    ;*IS NOT LISTED THE FLAG IS RESET.
8813                                    ;;*****************************************************************
8814  040146  104407                   BDSRCK: SAVREG
8815  040150  012737  004232  040162           MOV     #BSFACT,1$      ;SET TABLE TO SEARCH
8816  040156  004437  040004           2$:     JSR     R4,SRHTBS       ;GO SEARCH IT
8817  040162  000000                   1$:     .WORD                   ;TABLE ADDRESS GOES HERE
8818  040164  012603                           MOV     (SP)+,R3        ;GET NUMBER OF BAD SECTORS, IF ANY
8819  040166  001015                           BNE     6$              ;IF ANY, GO TEST WHICH ONES
8820  040170  023727  040162  003232  7$:     CMP     1$,#BSSOFT      ;ELSE TEST IF OTHER TABLE ALREADY SEARCHED
8821  040176  001404                           BEQ     3$              ;IF YES-EXIT
8822  040200  012737  003232  040162           MOV     #BSSOFT,1$      ;SET OTHER TABLE FOR SEARCH
8823  040206  000763                           BR      2$              ;GO SEARCH IT
8824  040210  042737  001000  005504  3$:     BIC     #BADSEC,RECODE  ;CLEAR BAD SECTOR BIT
8825  040216  104410                   4$:     RESREG
8826  040220  000207                           RTS     PC              ;RETURN
8827  040222  022602                   6$:     CMP     (SP)+,R2        ;THIS SECTOR IN TABLE?
8828  040224  001403                           BEQ     8$              ;YES-GO SET BIT & EXIT
8829  040226  005303                           DEC     R3              ;DECREMENT BAD SECTOR COUNT
8830  040230  001374                           BNE     6$              ;IF NOT ZERO-CHECK NEXT ENTRY
8831  040232  000756                           BR      7$              ;ELSE GO SEARCH OTHER TABLE
8832  040234  052737  001000  005504  8$:     BIS     #BADSEC,RECODE  ;SET BAD SECTOR BIT
8833  040242  005303                   9$:     DEC     R3              ;CLEAR STACK OF OTHER BAD SECTOR
8834  040244  001764                           BEQ     4$              ;NUMBER
8835  040246  005726                           TST     (SP)+
8836  040250  000774                           BR      9$              ;EXIT
8837
8838
8839
8840                                    ;;*****************************************************************
8841                                    .SBTTL  CALL DRIVER ROUTINE
8842                                    ;*ENTRY JSR PC,DRVCAL
8843                                    ;*              WITH R5 POINTING TO PARAMETER BLOCK
8844                                    ;*RETURN RTS PC
8845                                    ;*
8846                                    ;*THIS ROUTINE IS USED TO INITIATE A SUBSYSTEM OPERATION BY
8847                                    ;*CALLING THE DRIVER. THE PARAMETER BLOCK MUST BE SET UP
8848                                    ;*BY THE CALLER AND R5 MUST POINT TO THE PARAMETER
8849                                    ;*BLOCK WHEN THE ROUTINE IS CALLED.
8850                                    ;*
8851                                    ;*THIS ROUTINE WAITS FOR THE OPERATION TO BE COMPLETED.
8852                                    ;*WHILE WAITING THE WATCHDOG TIMER IS CALLED TO PREVENT
8853                                    ;*SILENT DEATH IN CASE THE SUBSYSTEM DOES NOT PROVIDE AN
8854                                    ;*INTERRUPT. THE TERMINATION HANDLER ROUTINES WILL SET THE DONE
8855                                    ;*FLAG WHICH KEYS THE ROUTINE TO RETURN TO THE CALLER.
8856                                    ;;*****************************************************************
8857  040252                           DRVCAL:
8858  040252  104407                           SAVREG                  ;SAVE R0-R5
8859  040254  004737  032426                   JSR     PC,STALL        ;PERFORM A STALL IF REQUIRED
8860  040260  004737  036266                   JSR     PC,CTLOUT       ;CHECK FOR (^C) OR (^Z) KBD INPUT
8861  040264  105037  003123                   CLRB    DONE            ;CLEAR DONE FLAG
8862  040270  105037  003137                   CLRB    DRNAFG          ;CLEAR DRIVE SIEZED BY OTHER PORT FLAG
8863  040274  004737  040542                   JSR     PC,STRCMD       ;STORE PREV AND CURRENT COMMANDS
8864  040300  105737  003136                   TSTB    DULACS          ;SEE IF DUAL-ACCESS TEST
8865  040304  001502                           BEQ     26$             ;BR IF NOT
8866  040306  005077  140632                   CLR     @$TKS           ;DISABLE KBD INPUT
```

```
 8867  040312  113762  005510  000010          MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
 8868  040320  012712  000001                  MOV     #BIT0,(R2)      ;SELECT THE DRIVE
 8869  040324  032712  000200          20$:    BIT     #BIT7,(R2)      ;WAIT FOR C. READY
 8870  040330  001775                          BEQ     20$
 8871  040332  032762  010000  000010          BIT     #BIT12,RKCS2(R2)        ;SEE IF NED ERROR SET
 8872  040340  001060                          BNE     28$             ;BR IF YES
 8873  040342  032762  000001  000012          BIT     #BIT0,RKDS(R2)  ;SEE IF DRIVE IS AVAILABLE
 8874  040350  001044                          BNE     30$             ;BR IF YES
 8875  040352  012712  100000                  MOV     #100000,(R2)    ;ISSUE CONTROLLER CLEAR
 8876  040356  113700  005510                  MOVB    DRIVE,R0        ;GET DRIVE NO.
 8877  040362  012703  000100                  MOV     #100,R3         ;SET OUTER COUNTER
 8878  040366  005001                  22$:    CLR     R1              ;INIT INNER COUNTER
 8879  040370  005201                  33$:    INC     R1              ;INCR INNER COUNTER
 8880  040372  001027                          BNE     22$
 8881  040374  005303                          DEC     R3              ;DECR OUTER COUNTER
 8882  040376  001373                          BNE     32$             ;BR IF NO TIME-OUT YET
 8883  040400  052737  000200  005504          BIS     #ABORT,RECODE   ;SET ABORT FLAG
 8884  040406  112737  000101  005233          MOVB    #SELDRV,PRVCMD+1 ;SET PREV CMND = SELECT DRIVE
 8885  040414  112765  000176  000001          MOVB    #CONCLR,P.CMND(R5) ;SET CURRENT CMND = CONTR CLEAR
 8886  040422  113737  005510  005232          MOVB    DRIVE,PRVCMD    ;SET DRIVE NO. FOR PREV. CMND
 8887  040430  011265  000016                  MOV     (R2),P.CS1(R5)  ;GET CURRENT CONTENTS OF RKCS1
 8888  040434  004737  050156                  JSR     PC,I.CST1       ;GO READ CONTROLLER REGS
 8889  040440  004737  041650                  JSR     PC,REPSUP       ;SET UP REGS FOR PRINTOUT
 8890  040444  104107                          ERROR   107             ;TYPE 'DRIVE SIEZED BY OTHER PORT'
 8891  040446  000137  043724                  JMP     ALLTRM          ;REPORT DRIVE SIEZED AND HALT
 8892  040452  136062  003102  000017  22$:    BITB    I.DRV(R0),RKASOF+1(R2) ;WAIT FOR DRIVE ATTENTION
 8893  040460  001743                          BEQ     33$             ;BR IF NO ATT'N YET
 8894  040462  113762  005510  000010  30$:    MOVB    DRIVE,RKCS2(R2) ;SET DRIVE NO. IN RKCS2
 8895  040470  012712  000005                  MOV     #5,(R2)         ;ISSUE DRIVE CLEAR
 8896  040474  032712  000200          24$:    BIT     #BIT7,(R2)      ;WAIT FOR CONTROLLER READY
 8897  040500  001775                          BEQ     24$
 8898  040502  012712  100000          28$:    MOV     #100000,(R2)    ;ISSUE CONTROLLER CLEAR
 8899  040506  004737  025766                  JSR     PC,PREPKB       ;ENABLE KBD INPUT
 8900  040512  010537  040522          26$:    MOV     R5,4$           ;GET PARAM BLOCK ADDRESS
 8901  040516  004737  050534                  JSR     PC,C.INIT       ;CALL DRIVER
 8902  040522  000000                  4$:     .WORD                   ;P.B. ADDRESS GOES HERE
 8903  040524  004737  045104          6$:     JSR     PC,W.WTCH       ;CALL WATCH DOG
 8904  040530  105737  003123                  TSTB    DONE            ;DONE SET?
 8905  040534  001773                          BEQ     6$              ;NO-LOOP
 8906  040536  104410                  12$:    RESREG                  ;RESTORE R0-R5
 8907  040540  000207                          RTS     PC              ;YES-RETURN
 8908
 8909
 8910
 8911                                  ;;*******************************************************************
 8912                                  ;*STRCMD - STORE PREVIOUS AND CURRENT SUBSYSTEM COMMANDS
 8913                                  ;;*******************************************************************
 8914  040542  104407                  STRCMD: SAVREG                  ;SAVE R0-R5
 8915  040544  012701  005232                  MOV     #PRVCMD,R1      ;ADDR OF PREV COMMAND STORAGE
 8916  040550  012700  005246                  MOV     #COMSTR,R0      ;ADDR OF CURRENT COMMAND STORAGE
 8917                                  ;STORE PREVIOUS COMMAND
 8918  040554  012021                          MOV     (R0)+,(R1)+
 8919  040556  012021                          MOV     (R0)+,(R1)+
 8920  040560  012021                          MOV     (R0)+,(R1)+
 8921  040562  012021                          MOV     (R0)+,(R1)+
 8922  040564  012021                          MOV     (R0)+,(R1)+
```

```
 8923  040566  012021                          MOV     (R0)+,(R1)+
 8924  040570  105737  003144                  TSTB    UBMPRS          ;SEE IF UNIBUS MAP PRESENT
 8925  040574  001414                          BEQ     4$              ;BR IF NOT
 8926  040576  013737  005270  005264          MOV     CRMPH0,PRMPH0   ;STORE PREV U.B. MAP REG 0
 8927  040604  013737  005266  005262          MOV     CRMPL0,PRMPL0
 8928  040612  013737  170202  005270          MOV     @#MAPH00,CRMPH0 ;STORF CURRENT U.B. MAP REG 0
 8929  040620  013737  170200  005266          MOV     @#MAPL00,CRMPI 0
 8930                                ;STORE CURRENT COMMAND
 8931  040626  012701  005246              4$: MOV     #COMSTR,R1
 8932  040632  012521                          MOV     (R5)+,(R1)+     ;R5 POINTS TO DRIVER PARAM BLK
 8933  040634  012521                          MOV     (R5)+,(R1)+
 8934  040636  012521                          MOV     (R5)+,(R1)+
 8935  040640  012521                          MOV     (R5)+,(R1)+
 8936  040642  012521                          MOV     (R5)+,(R1)+
 8937  040644  012521                          MOV     (R5)+,(R1)+
 8938  040646  104410                          RESREG                  ;RESTORE R0-R5
 8939  040650  000207                          RTS     PC              ;RETURN
 8940
 8941
 8942
 8943                                ;;**************************************************************
 8944                                .SBTTL  DRIVE ERROR FREE RETURN ROUTINE
 8945                                ;*THIS ROUTINE IS CALLED BY THE DRIVER WHEN NO ERROR
 8946                                ;*HAS BEEN DETECTED IN THE OPERATION. THE ROUTINE SETS THE
 8947                                ;*DONE FLAG THAT IS TESTED IN THE DRIVER CALLING
 8948                                ;*ROUTINE.
 8949                                ;;**************************************************************
 8950  040652  152737  000377  003123 ERRFRE: BISB    #377,DONE       ;SET THE DONE FLAG
 8951  040660  032737  100000  005504          BIT     #ANYDER,RECODE  ;TEST IF ANY DATA ERROR
 8952  040666  001403                          BEQ     2$              ;IF NO - DO ERROR RECOVERY PRINT TEST
 8953  040670  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 8954  040674  000413                          BR      1$              ;EXIT
 8955  040676  105737  003126              2$: TSTB    ERRCNT          ;CHECK IF ANY ERRORS HAVE OCCURRED
 8956  040702  001410                          BEQ     1$              ;NO - SKIP TO EXIT
 8957  040704  005037  001174                  CLR     $REG5
 8958  040710  113737  003126  001174          MOVB    ERRCNT,$REG5    ;GET RETRY COUNT
 8959  040716  104101                          ERROR   101             ;PRINT RETRY SUCCESSFUL MESSAGE
 8960  040720  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 8961  040724  005037  005504              1$: CLR     RECODE          ;CLEAR RECOVERY FLAGS
 8962  040730  000207                          RTS     PC              ;RETURN
 8963                                ;;**************************************************************
 8964                                .SBTTL  TYPE ERROR ROUTINE
 8965                                ;*ENTRY -       JSR     PC,TYPERR
 8966                                ;*RETURN -      RTS     PC
 8967                                ;*
 8968                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 8969                                ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
 8970                                ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
 8971                                ;*THE ERROR.
 8972                                ;;**************************************************************
 8973  040732  104407                TYPERR: SAVREG
 8974  040734  105237  003130                  INCB    DRVERS          ;INCR ERROR COUNT FOR THIS DRIVE
 8975  040740  032737  000040  005770          BIT     #BIT5,CS        ;SEE IF DRIVE SHOULD BE DROPPED
 8976                                                                  ;  IF ERROR LIMIT EXCEEDED
 8977  040746  001412                          BEQ     9$              ;BR IF NOT
 8978  040750  123727  003130  000024          CMPB    DRVERS,#20.     ;SEE IF 20(DEC) ERRORS EXCEEDED
```

```
 8979  040756  002406                        BLT     9$              ;BR IF NOT
 8980  040760  105037  003130                CLRB    DRVERS          ;CLEAR DRIVE ERROR COUNT
 8981  040764  104401  011251                TYPE    ,DROPDR         ;TYPE 'DROPPING DRIVE'
 8982  040770  000137  016754                JMP     NEWDRV          ;PROCEED TO TEST NEXT DRIVE
 8983  040774  032777  020000  140136  9$:   BIT     #SW13,@SWR      ;INHIBIT ERROR TYPEOUTS?
 8984  041002  001402                        BEQ     6$              ;BR IF NO
 8985  041004  000137  041410                JMP     20$
 8986  041010  005000                  6$:   CLR     R0              ;CLR R0 FOR ERROR NUMBER
 8987  041012  005005                        CLR     R5              ;INIT INDENT INDICATOR
 8988  041014  005105                        COM     R5
 8989  041016  113700  001114                MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
 8990  041022  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
 8991  041024  006300                        ASL     R0
 8992  041026  006300                        ASL     R0
 8993  041030  006300                        ASL     R0
 8994  041032  062700  001400            1$: ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
 8995  041036  012037  041056                MOV     (R0)+,2$        ;GET EM POINTER
 8996  041042  001406                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 8997  041044  104401  012404                TYPE    ,CR2LF
 8998  041050  104401  056320                TYPE    ,AS2SP2         ;TYPE "** "
 8999  041054  104401                        TYPE                    ;TYPE ERROR MESSAGE (EM)
 9000  041056  000000                  2$:   .WORD   0               ;EM POINTER GOES HERE
 9001  041060  012037  041234            3$: MOV     (R0)+,4$        ;GET DH POINTER
 9002  041064  001467                        BEQ     5$              ;BR IF THERE ISN'T ONE
 9003  041066  104401  001315                TYPE    ,$CRLF
 9004  041072  104401  056306                TYPE    ,TSTMSG         ;TYPE "    TEST "
 9005  041076  013746  001102                MOV     $TSTNM,-(SP)    ;GET TEST NO. ON STACK
 9006  041102  104403                        TYPOS                   ;TYPE IT
 9007  041104     002                        .BYTE   2               ;2 DIGITS
 9008  041105     000                        .BYTE   0               ;SUPPRESS LEADING ZEROS
 9009  041106  104401  012404                TYPE    ,CR2LF
 9010  041112  032777  010000  140020        BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 9011  041120  001133                        BNE     20$             ;BR IF YES
 9012  041122  104401  060767                TYPE    ,DH105          ;TYPE "PREVIOUS COMMAND :"
 9013  041126  104401  001315                TYPE    ,$CRLF
 9014  041132  104401  061132                TYPE    ,DH101+10       ;TYPE PREV COMMAND HEADER
 9015  041136  104401  001315                TYPE    ,$CRLF
 9016  041142  012701  000006                MOV     #6,R1           ;SIX COMMAND VALUES
 9017  041146  012702  001236                MOV     #$REG26,R2      ;STARTING ADDR OF PREV CMND VALUES
 9018  041152  012246                  30$:  MOV     (R2)+,-(SP)     ;PUT A WORD ON STACK
 9019  041154  104402                        TYPOC                   ;TYPE IT
 9020  041156  104401  012420                TYPE    ,SPACE2         ;TYPE SEPARATORS
 9021  041162  005301                        DEC     R1              ;SEE IF 7 VALUES TYPED YET
 9022  041164  001372                        BNE     30$             ;BR IF NOT
 9023  041166  104401  001315                TYPE    ,$CRLF
 9024  041172  104401  012420                TYPE    ,SPACE2         ;INDENT
 9025  041176  104401  061211                TYPE    ,DH102          ;TYPE HDR FOR BA DATA
 9026  041202  104401  001315                TYPE    ,$CRLF
 9027  041206  104401  012420                TYPE    ,SPACE2         ;INDENT
 9028  041212  012246                        MOV     (R2)+,-(SP)
 9029  041214  104402                        TYPOC                   ;TYPE PREV. HI BA BITS
 9030  041216  104401  012420                TYPE    ,SPACE2
 9031  041222  011246                        MOV     (R2),-(SP)
 9032  041224  104402                        TYPOC                   ;TYPE PREV. LO BA BITS
 9033  041226  104401  012404                TYPE    ,CR2LF
 9034  041232  104401                        TYPE                    ;TYPE DATA HEADER
```

```
 9035  041234  000000                  4$:     .WORD   0               ;DH POINTER GOES HERE
 9036  041236  104401  001315                  TYPE    ,$CRLF
 9037  041242  005005                          CLR     R5              ;INIT INDENT INDICATOR
 9038  041244  032777  010000  137666  5$:     BIT     #BIT12,@SWR     ;REPORT DESCRIPTION ONLY ?
 9039  041252  001056                          BNE     20$             ;BR IF YES
 9040  041254  012001                          MOV     (R0)+,R1        ;GET DT POINTER
 9041  041256  001454                          BEQ     20$             ;BRANCH IF THERE ARE NONE
 9042  041260  012000                          MOV     (R0)+,R0        ;GET DF POINTER
 9043  041262  012002                          MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
 9044  041264  112003                  10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
 9045  041266  105720                          TSTB    (R0)+           ;BUMP PAST FORMAT WORD
 9046  041270  005703                          TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
 9047  041272  001417                          BEQ     14$             ;NO - SKIP DATA PRINT
 9048  041274  005705                          TST     R5              ;SEE IF SHOULD INDENT
 9049  041276  001002                          BNE     11$             ;BR IF NOT
 9050  041300  104401  012420                  TYPE    ,SPACE2         ;INDENT
 9051  041304  013146                  11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
 9052  041306  104402                          TYPOC                   ;TYPE IT
 9053  041310  005303                          DEC     R3              ;MORE DATA WORDS
 9054  041312  001403                          BEQ     12$             ;NO-BRANCH
 9055  041314  104401  012420                  TYPE    ,SPACE2         ;TYPE SEPARATORS
 9056  041320  000771                          BR      11$             ;LOOP
 9057  041322  005702                  12$:    TST     R2              ;SEE IF <CR>,<LF> NEEDED
 9058  041324  001402                          BEQ     14$             ;BR IF NOT
 9059  041326  104401  001315                  TYPE    ,$CRLF          ;TYPE IT
 9060  041332  005302                  14$:    DEC     R2              ;MORE DH'S?
 9061  041334  003425                          BLE     20$             ;NO-BRANCH
 9062  041336  012037  041370          15$:    MOV     (R0)+,16$       ;GET NEXT DH POINTER
 9063  041342  105710                          TSTB    (R0)            ;SEE IF ANY DATA FOR HDR
 9064  041344  001004                          BNE     34$             ;BR IF YES
 9065  041346  104401  001315                  TYPE    ,$CRLF          ;SKIP EXTRA LINE
 9066  041352  005005                          CLR     R5              ;RE-INIT INDENT INDICATOR
 9067  041354  000404                          BR      36$
 9068  041356  005105                  34$:    COM     R5              ;COMPLEMENT INDENT INDICATOR
 9069  041360  001002                          BNE     36$             ;BR IF NO INDENT REQUIRED
 9070  041362  104401  012420                  TYPE    ,SPACE2         ;INDENT
 9071  041366  104401                  36$:    TYPE                    ;TYPE DH
 9072  041370  000000                  16$:    .WORD   0               ;DH POINTER GOES HERE
 9073  041372  104401  001315                  TYPE    ,$CRLF
 9074  041376  105710                          TSTB    (R0)            ;TYPE A DT?
 9075  041400  001331                          BNE     10$             ;YES-BRANCH
 9076  041402  062700  000002                  ADD     #2,R0           ;INCREMENT DF POINTER
 9077  041406  000751                          BR      14$             ;SEE IF END OF DF BLOCK
 9078  041410  104410                  20$:    RESREG
 9079  041412  000207                          RTS     PC
 9080                          ;;*****************************************************************
 9081                          .SBTTL  CONTROLLER ERROR REPORTER ROUTINE
 9082                          ;*ENTRY:        JSR PC, CONERR
 9083                          ;*RETURN:       RTS PC
 9084                          ;*
 9085                          ;*THIS ROUTINE DECODES THE CONTROLLER ERROR WORD AND
 9086                          ;*REPORTS THE APPROPRIATE MESSAGE. THE RK611 REGISTERS ARE
 9087                          ;*RETRIEVED FROM THE RK611 AND PLACED IN THE PARAMETER
 9088                          ;*BLOCK. THIS IS DONE BECAUSE PARM 0 MAY NOT BE VALID
 9089                          ;*AT THIS TIME.
 9090                          ;;*****************************************************************
```

```
 9091  041414  104407                    CONERR: SAVREG                  ;SAVE R0-R5
 9092  041416  152737  000377  003123            BISB    #377,DONE       ;SET DONE FLAG
 9093  041424  105237  003126                    INCB    ERRCNT          ;INCREMENT ERROR COUNT
 9094  041430  004737  044316                    JSR     PC,TOPROC       ;LOAD RK REGS INTO $REGS
 9095  041434  032737  000001  003052            BIT     #BIT0,E.CONT    ;ERROR 0?
 9096  041442  001402                            BEQ     1$              ;NO-BRANCH
 9097  041444  104064                            ERROR   64              ;CLEAR CONT DID NOT CLEAR ERROR
 9098  041446  000474                            BR      7$
 9099  041450  032737  000002  003052  1$:       BIT     #BIT1,E.CONT    ;ERROR 1?
 9100  041456  001402                            BEQ     2$              ;NO-BRANCH
 9101  041460  104065                            ERROR   65              ;NO ATTENTION IN ATTENTION SUM REG
 9102  041462  000466                            BR      7$
 9103  041464  032737  000004  003052  2$:       BIT     #BIT2,E.CONT    ;ERROR 2?
 9104  041472  001407                            BEQ     3$              ;NO-BRANCH
 9105  041474  105737  003137                    TSTB    DRNAFG          ;SEE IF DRIVE WAS SIEZED BY OTHER PORT
 9106  041500  001402                            BEQ     15$             ;BR IF NOT
 9107  041502  105237  003140                    INCB    REISSU          ;SET FLAG TO RE-ISSUE COMMAND
 9108  041506  104066                    15$:    ERROR   66              ;UNSOLICITED ATTENTION
 9109  041510  000453                            BR      7$
 9110  041512  032737  000010  003052  3$:       BIT     #BIT3,E.CONT    ;ERROR 3?
 9111  041520  001402                            BEQ     4$              ;NO-BRANCH
 9112  041522  104067                            ERROR   67              ;UNEXPECTED DATA TYPE ERROR
 9113  041524  000445                            BR      7$
 9114  041526  032737  000020  003052  4$:       BIT     #BIT4,E.CONT    ;ERROR 4?
 9115  041534  001402                            BEQ     5$              ;NO-BRANCH
 9116  041536  104070                            ERROR   70              ;ATTENTION DID NOT RESET WITH CLEAR
 9117  041540  000437                            BR      7$
 9118  041542  032737  000040  003052  5$:       BIT     #BIT5,E.CONT    ;ERROR 5?
 9119  041550  001402                            BEQ     6$              ;NO-BRANCH
 9120  041552  104071                            ERROR   71              ;SUBSYS CLEAR DIDN'T CLEAR DRIVE ATTENTION
 9121  041554  000431                            BR      7$
 9122  041556  032737  000400  003052  6$:       BIT     #BIT8,E.CONT
 9123  041564  001402                            BEQ     8$
 9124  041566  104072                            ERROR   72              ;DATA LATE WHEN UNLOADING HEADER
 9125  041570  000423                            BR      7$
 9126  041572  032737  001000  003052  8$:       BIT     #BIT9,E.CONT
 9127  041600  001402                            BEQ     9$
 9128  041602  104073                            ERROR   73              ;CONTROLLER ERROR DURING DRIVER SERVICE
 9129  041604  000415                            BR      7$
 9130  041606  032737  002000  003052  9$:       BIT     #BIT10,E.CONT
 9131  041614  001402                            BEQ     10$
 9132  041616  104074                            ERROR   74              ;DRIVE DETECTED PARITY ERROR
 9133  041620  000407                            BR      7$
 9134  041622  032737  100000  003052  10$:      BIT     #BIT15,E.CONT
 9135  041630  001402                            BEQ     11$
 9136  041632  104052                            ERROR   52              ;MULTIPLE DRIVE SELECT
 9137  041634  000401                            BR      7$
 9138  041636  104075                    11$:    ERROR   75              ;UNDEFINED ERROR
 9139  041640  005037  003052            7$:     CLR     E.CONT          ;CLEAR CONTROLLER ERROR WORD
 9140  041644  000137  044116                    JMP     BGNRTY          ;GO DO RETRY
 9141                                    ;;*****************************************************************
 9142                                    .SBTTL  REPORT SUPPORT ROUTINE
 9143                                    ;*THIS ROUTINE LOADS ALL THE PARAMETER BLOCK DATA TO BE REPORTED
 9144                                    ;*INTO THE PROPER TEMPORARY REGISTERS FOR REPORTING.  ALL THESE MAY
 9145                                    ;*NOT BE INCLUDED IN THE REPORT, BUT THEY ARE LOADED ANYWAY.
 9146  041650                            REPSUP:
```

```
 9147  041650  104407                        SAVREG
 9148  041652  005037  005506                CLR     ERRCOM          ;CLEAR ERROR COMMAND STORE
 9149  041656  116537  000001  005506        MOVB    P.CMND(R5),ERRCOM ;STORE COMMAND START VALUES
 9150  041664  012700  001162                MOV     #$REG0,R0       ;FOR REPORTING
 9151  041670  016520  000002                MOV     P.CYLN(R5),(R0)+
 9152  041674  116520  000005                MOVB    P.TRCK(R5),(R0)+
 9153  041700  105020                        CLRB    (R0)+
 9154  041702  116520  000004                MOVB    P.SECT(R5),(R0)+
 9155  041706  105020                        CLRB    (R0)+
 9156  041710  016520  000012                MOV     P.WC(R5),(R0)+
 9157  041714  012700  001174                MOV     #$REG5,R0
 9158  041720  116503  000007                MOVB    P.BAHI(R5),R3
 9159  041724  042703  177774                BIC     #177774,R3
 9160  041730  010337  001256                MOV     R3,$REG36       ;HI BA BITS
 9161  041734  016537  000010  001260        MOV     P.BALO(R5),$REG37 ;LO BA BITS
 9162  041742  016520  000016                MOV     P.CS1(R5),(R0)+ ;GET ALL THE VALUES FROM THE
 9163  041746  016520  000020                MOV     P.CS2(R5),(R0)+ ;PARAMETER BLOCK AND LOAD
 9164  041752  016520  000030                MOV     P.DCYL(R5),(R0)+ ;THE TEMPORARY REGISTERS
 9165  041756  016520  000026                MOV     P.DTS(R5),(R0)+ ;FOR REPORTING. ALL THIS
 9166  041762  016520  000022                MOV     P.WCR(R5),(R0)+ ;DATA MAY NOT BE VALID
 9167                                                                ;FOR ALL REPORTS (TO BE
 9168  041766  016520  000024                MOV     P.BAR(R5),(R0)+ ;DETERMINED LATER) BUT IT IS
 9169  041772  016520  000032                MOV     P.ASOF(R5),(R0)+ ;STORED ANY WAY.
 9170  041776  016520  000036                MOV     P.DS(R5),(R0)+
 9171  042002  016520  000034                MOV     P.ER(R5),(R0)+
 9172  042006  016520  000040                MOV     P.A00(R5),(R0)+
 9173  042012  016520  000042                MOV     P.B00(R5),(R0)+
 9174  042016  016520  000044                MOV     P.A01(R5),(R0)+
 9175  042022  016520  000046                MOV     P.B01(R5),(R0)+
 9176  042026  016520  000050                MOV     P.A10(R5),(R0)+
 9177  042032  016520  000052                MOV     P.B10(R5),(R0)+
 9178  042036  016520  000054                MOV     P.A11(R5),(R0)+
 9179  042042  016520  000056                MOV     P.B11(R5),(R0)+
 9180                              ;STORE PREVIOUS COMMAND FOR PRINTOUT
 9181  042046  012701  001236                MOV     #$REG26,R1      ;ADRS OF PRINT BUF AREA
 9182  042052  012700  005232                MOV     #PRVCMD,R0      ;ADRS OF PREV CMND STORAGE
 9183  042056  112021                        MOVB    (R0)+,(R1)+     ;DRIVE NO.
 9184  042060  105021                        CLRB    (R1)+
 9185  042062  112021                        MOVB    (R0)+,(R1)+     ;COMMAND
 9186  042064  105021                        CLRB    (R1)+
 9187  042066  012021                        MOV     (R0)+,(R1)+     ;CYL ADDRESS
 9188  042070  116021  000001                MOVB    1(R0),(R1)+     ;TRACK
 9189  042074  105021                        CLRB    (R1)+
 9190  042076  111021                        MOVB    (R0),(R1)+      ;SECTOR
 9191  042100  105021                        CLRB    (R1)+
 9192  042102  016021  000006                MOV     6(R0),(R1)+     ;WORD COUNT
 9193  042106  116003  000003                MOVB    3(R0),R3        ;HI BA BITS
 9194  042112  042703  177774                BIC     #177774,R3
 9195  042116  010321                        MOV     R3,(R1)+
 9196  042120  016011  000004                MOV     4(R0),(R1)      ;LO BA BITS
 9197  042124  104410                        RESREG
 9198  042126  000207                        RTS     PC
 9199                              ;;*****************************************************************
 9200                              .SBTTL  REPORT ERROR ROUTINE
 9201                              ;* ENTRY         JSR     PC, ERRHDL
 9202                              ;*RETURN        RTS     PC
```

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

042130  104407                  ERRHDL: SAVREG
042132  152737  000377  003123          BISB    #377,DONE       ;SET DONE FLAG
042140  105237  003126          ERRCNT  INCB    ERRCNT          ;INCREMENT ERROR COUNT
042144  005037  005504          RECODE  CLR     RECODE          ;CLEAR RECOVERY CODE WORD
042150  032737  000400  005504  ER2ENT: BIT     #LEV2ER,RECODE  ;TEST IF 2ND LEVEL ERROR
042156  001402                          BEQ     52$             ;NO - SKIP PARAM BLOCK CHANGE
042160  012705  002714                  MOV     #PARM1,R5       ;ELSE SET R5 TO PARAMETER BLOCK 1
042164  012737  044270  003046  52$:    MOV     #RETANL,A.ABNL  ;SET NOW ABNORMAL AND NORMAL RETURN FOR
042172  012737  044306  003044          MOV     #RETNML,A.NORM  ;DRIVER OPERATIONS IN ERROR PROCESSING
042200  004737  041650                  JSR     PC,REPSUP       ;GO SET UP REGISTERS FOR REPOPI
                                                                ;NOW BEGIN TESTING THE ERROR
                                                                ;BITS. THE SEQUENCE IN
                                                                ;WHICH THEY ARE TESTED IS
                                                                ;CONSIDERED SIGNIFICANT IN
                                                                ;THAT ERRORS OF A MORE
                                                                ;CATASTROPHIC NATURE ARE FIRST
                                                                ;TESTED.
                                                                ;
                                                                ;IF AN ERROR IS FOUND SET,
                                                                ;THAT ERROR IS REPORTED AND
                                                                ;THE REPORTING IS TERMINATED.
                                                                ;IF ADDITIONAL ERRORS ARE SET,
                                                                ;THE RK611 REGISTER PRINTOUTS
                                                                ;WILL SHOW THIS BUT THE
                                                                ;REGISTER CONTENTS MUST BE
                                                                ;MANUALLY DECODED TO LOCATE THE
                                                                ;SECOND ERROR
042204  016504  000034                  MOV     P.ER(R5),R4     ;SET R4 TO ERROR REGISTER
042210  032765  020000  000020          BIT     #UPE,P.CS2(R5)  ;TEST UPE. IF UES-SET
042216  001406                          BEQ     1$              ;REPORT ERROR
042220  104001                          ERROR   1
042222  052737  000200  005504          BIS     #ABORT,RECODE
042230  000137  043624                  JMP     37$
042234  032765  004000  000020  1$:     BIT     #NEM,P.CS2(R5)  ;TEST NON-EXISTANT MEMORY
042242  001406                          BEQ     2$
042244  104002                          ERROR   2
042246  052737  000200  005504          BIS     #ABORT,RECODE
042254  000137  043624                  JMP     37$
042260  032765  010000  000020  2$:     BIT     #NED,P.CS2(R5)  ;TEST NON-EXISTANT DRIVE
042266  001412                          BEQ     3$
042270  032765  000400  000020          BIT     #UFE,P.CS2(R5)  ;TEST IF NED & UFE BOTH SET
042276  001403                          BEQ     38$
042300  104035                          ERROR   35              ;NED & UFE BOTH SET ERROR
042302  000137  043624                  JMP     37$
042306  104003                  38$:    ERROR   3               ;NED ONLY
042310  000137  043624                  JMP     37$
042314  032765  000400  000020  3$:     BIT     #UFE,P.CS2(R5)  ;TEST UNIT FIELD ERROR
042322  001412                          BEQ     4$
042324  032765  010000  000020          BIT     #NED,P.CS2(R5)  ;TEST IF UFE & NED BOTH SET
042332  001403                          BEQ     39$             ;NO-SKIP
```

```
 9259  042334  104035                        ERROR   35              ;REPORT NED & UFE BOTH SET
 9260  042336  000137  043624                JMP     37$
 9261  042342  104004                39$:    ERROR   4               ;REPORT UFE ONLY
 9262  042344  000137  043624                JMP     37$
 9263  042350  032765  000100  000014  4$:    BIT     #CMDTO,P.PRST(R5) ;
 9264  042356  001423                        BEQ     5$
 9265  042360  004737  044316                JSR     PC,TOPROC       ;GO PROCESS TIMEOUT
 9266  042364  032737  002000  005504        BIT     #TWOTOS,RECODE  ;2ND TIMEOUT IN TIMEOUT PROC?
 9267  042372  001007                        BNE     40$             ;YES - SKIP TO ERROR 56
 9268  042374  032737  000400  005504        BIT     #LEV2ER,RECODE  ;TEST IF LEVEL 2 ERROR
 9269  042402  001006                        BNE     41$             ;YES - SKIP TO ERROR 57
 9270  042404  104005                        ERROR   5               ;ELSE MAKE FULL TIMEOUT REPORT
 9271  042406  000137  043624                JMP     37$
 9272  042412  104056                40$:    ERROR   56              ;TWO TIMEOUTS ERROR REPORT
 9273  042414  000137  043624                JMP     37$
 9274  042420  104057                41$:    ERROR   57              ;2ND LEVEL ERROR REPORT
 9275  042422  000137  042150                JMP     ER2ENT          ;GO BUILD AND MAKE 2ND REPORT
 9276  042426  032765  010000  000014  5$:    BIT     #DRVSZD,P.PRST(R5) ;SEE IF DRIVE SIEZED BY OTHER PORT
 9277  042434  001403                        BEQ     65$             ;BR IF DRIVE IS AVAILABLE
 9278  042436  104107                        ERROR   107             ;DRIVE SIEZED BY OTHER PORT
 9279  042440  000137  043624                JMP     37$
 9280  042444  032765  020000  000016  65$.   BIT     #SPAR,P.CS1(R5) ;TEST D TO C PARITY ERROR
 9281  042452  001406                        BEQ     6$
 9282  042454  104006                        ERROR   6
 9283  042456  052737  004000  005504        BIS     #RCLREQ,RECODE
 9284  042464  000137  043624                JMP     37$
 9285  042470  032704  000010          6$:   BIT     #DRPAR,R4       ;TEST DRIVE DETECTED PARITY ERROR
 9286  042474  001406                        BEQ     7$
 9287  042476  104007                        ERROR   7
 9288  042500  052737  004000  005504        BIS     #RCLREQ,RECODE
 9289  042506  000137  043624                JMP     37$
 9290  042512  032765  000010  000036  7$:    BIT     #ACLO,P.DS(R5)  ;TEST AC LOW
 9291  042520  001403                        BEQ     8$
 9292  042522  104010                        ERROR   10
 9293  042524  000137  043624                JMP     37$
 9294  042530  032765  000020  000036  8$:    BIT     #SPDLSS,P.DS(R5) ;TEST SPEED LOSS
 9295  042536  001403                        BEQ     24$
 9296  042540  104011                        ERROR   11
 9297  042542  000137  043624                JMP     37$
 9298  042546  032765  004000  000016  24$:   BIT     #CTO,P.CS1(R5)  ;TEST FOR CONTROLLER TIMEOUT
 9299  042554  001401                        BEQ     25$
 9300  042556  104027                        ERROR   27
 9301  042560  032704  000001          25$:   BIT     #ILC,R4         ;TEST ILLEGAL FUNCTION CODE
 9302  042564  001403                        BEQ     10$
 9303  042566  104012                        ERROR   12
 9304  042570  000137  043624                JMP     37$
 9305  042574  032765  002000  000020  10$:   BIT     #PGE,P.CS2(R5)  ;TEST PROGRAMMING ERROR
 9306  042602  001403                        BEQ     11$
 9307  042604  104013                        ERROR   13
 9308  042606  000137  043624                JMP     37$
 9309  042612  032704  000004          11$:   BIT     #ILF,R4         ;TEST ILLEGAL DRIVE FUNCTION
 9310  042616  001403                        BEQ     12$
 9311  042620  104014                        ERROR   14
 9312  042622  000137  043624                JMP     37$
 9313  042626  032704  000040          12$:   BIT     #DTYE,R4        ·TEST DRIVE TYPE ERROR
 9314  042632  001403                        BEQ     13$
```

```
 9315  042634  104015                        ERROR   15
 9316  042636  000137  043624                JMP     37$
 9317  042642  032704  000020        13$:    BIT     #FMTE,R4        ;TEST FORMAT ERROR
 9318  042646  001403                        BEQ     14$
 9319  042650  104016                        ERROR   16
 9320  042652  000137  043624                JMP     37$
 9321  042656  032704  004000        14$:    BIT     #WLE,R4         ;TEST WRITE LOCK ERROR
 9322  042662  001403                        BEQ     15$
 9323  042664  104017                        ERROR   17
 9324  042666  000137  043624                JMP     37$
 9325  042672  032704  040000        15$:    BIT     #UNS,R4         ;TEST DRIVE UNSAFE
 9326  042676  001406                        BEQ     16$
 9327  042700  104020                        ERROR   20
 9328  042702  052737  000200  005504        BIS     #ABORT,RECODE
 9329  042710  000137  043724                JMP     ALLTRM
 9330  042714  032704  000002        16$:    BIT     #SKI,R4         ;TEST SEEK INCOMPLETE
 9331  042720  001406                        BEQ     17$
 9332  042722  104021                        ERROR   21
 9333  042724  052737  004000  005504        BIS     #RCLREQ,RECODE
 9334  042732  000137  043624                JMP     37$
 9335  042736  032704  001000        17$:    BIT     #COE,R4         ;TEST CYLINDER OVERFLOW
 9336  042742  001406                        BEQ     18$
 9337  042744  104022                        ERROR   22
 9338  042746  052737  004000  005504        BIS     #RCLREQ,RECODE
 9339  042754  000137  043624                JMP     37$
 9340  042760  032704  002000        18$:    BIT     #IDAE,R4        ;TEST ILLEGAL CYLINDER
 9341  042764  001406                        BEQ     19$
 9342  042766  104023                        ERROR   23
 9343  042770  052737  004000  005504        BIS     #RCLREQ,RECODE
 9344  042776  000137  043624                JMP     37$
 9345  043002  032765  000040  000036  19$:   BIT     #DROT,P.DS(R5)  ;TEST DRIVE OFF TRACK
 9346  043010  001406                        BEQ     20$
 9347  043012  104024                        ERROR   24
 9348  043014  052737  004000  005504        BIS     #RCLREQ,RECODE
 9349  043022  000137  043624                JMP     37$
 9350  043026  032704  010000        20$:    BIT     #DTE,R4         ;TEST DRIVE TIMING ERROR
 9351  043032  001406                        BEQ     21$
 9352  043034  104025                        ERROR   25
 9353  043036  052737  000200  005504        BIS     #ABORT,RECODE
 9354  043044  000137  043624                JMP     37$
 9355  043050  032765  100000  000020  21$:   BIT     #DLT,P.CS2(R5)  ;TEST DATA LATE
 9356  043056  001403                        BEQ     22$
 9357  043060  104026                        ERROR   26
 9358  043062  000137  043624                JMP     37$
 9359  043066  032704  020000        22$:    BIT     #OPI,R4         ;TEST IF OPI ERROR
 9360  043072  001457                        BEQ     29$
 9361  043074  052737  000010  005504        BIS     #OPIERR,RECODE
 9362  043102  105737  003121                TSTB    DERCNT          ;TEST IF FIRST ERROR
 9363  043106  001402                        BEQ     50$
 9364  043110  000137  043624                JMP     37$             ;NO - SKIP REPORT
 9365  043114  032737  000400  005504  50$:   BIT     #LEV2ER,RECODE  ;TEST IF A SECOND LEVEL 2 ERROR
 9366  043122  001403                        BEQ     26$             ;HAS ALREADY OCCURRED
 9367  043124  104054                27$:    ERROR   54
 9368  043126  000137  043624                JMP     37$             ;GET OUT OF ERROR REPORT
 9369  043132  004737  044756        26$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
 9370  043136  004737  044466                JSR     PC,RDHD0        ;GET HEADER OF SECTOR 0
```

```
 9371  043142  032737  000400  005504          BIT     #LEV2ER,RECODE  ;TEST IF ERROR GETTING HDR
 9372  043150  001025                          BNE     28$             ;IF YES-GO MAKE ABREVIATED REPORT
 9373  043152  013702  003036                  MOV     RKBAS,R2        ;STORE HEADER 0 INTO REGISTERS
 9374  043156  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;RESET INTERRUPT ENABLE
 9375  043164  016237  000024  001202          MOV     RKDB(R2),$REG10 ;FOR REPORTING
 9376  043172  016237  000024  001204          MOV     RKDB(R2),$REG11
 9377  043200  016237  000024  001206          MOV     RKDB(R2),$REG12
 9378  043206  032762  100000  000000          BIT     #CERR,RKCS1(R2) ;TEST IF ERROR DURING STORAGE
 9379  043214  001343                          BNE     27$
 9380  043216  104030                          ERROR   30              ;MAKE OPI REPORT
 9381  043220  000137  043624                  JMP     37$
 9382  043224  104054                  28$:    ERROR   54
 9383  043226  000137  042150                  JMP     ER2ENT          ;GO MAKE 2ND LEVEL REPORT
 9384  043232  032704  000400          29$:    BIT     #HVRC,R4        ;TEST IF HVRC ERROR
 9385  043236  001457                          BEQ     23$
 9386  043240  052737  000004  005504          BIS     #HVRCER,RECODE
 9387  043246  105737  003121                  TSTB    DERCNT          ;TEST IF FIRST ERROR
 9388  043252  001402                          BEQ     30$             ;YES - REPORT
 9389  043254  000137  043624                  JMP     37$             ;JUMP TO RETURN
 9390  043260  032737  000400  005504  30$:    BIT     #LEV2ER,RECODE  ;TEST IF A 2ND LEVEL ERROR HAS ALREADY
 9391  043266  001403                          BEQ     31$             ;OCCURRED. NO-SKIP EXIT
 9392  043270  104055                  51$:    ERROR   55
 9393  043272  000137  043624                  JMP     37$             ;GET OUT OF ERROR REPORT
 9394  043276  004737  044756          31$:    JSR     PC,BLDEXH       ;GO BUILD EXPECTED HEADER
 9395  043302  004737  044466                  JSR     PC,RDHD0        ;GO GET HDR 0
 9396  043306  032737  000400  005504          BIT     #LEV2ER,RECODE  ;TEST IF ERROR IN GETTING HDR
 9397  043314  001025                          BNE     32$             ;IF YES-GO MAKE ABREVIATED REPORT
 9398  043316  013702  003036                  MOV     RKBAS,R2        ;GET RK611 BASE ADDRESS
 9399  043322  042762  000100  000000          BIC     #IE,RKCS1(R2)   ;CLEAR INTERRUPT ENABLE
 9400  043330  016237  000024  001202          MOV     RKDB(R2),$REG10 ;STORE OFF HEADER
 9401  043336  016237  000024  001204          MOV     RKDB(R2),$REG11
 9402  043344  016237  000024  001206          MOV     RKDB(R2),$REG12
 9403  043352  032762  100000  000000          BIT     #CERR,RKCS1(R2) ;TEST IN ANY ERROR IN UNLOAD
 9404  043360  001343                          BNE     51$             ;IY YES - GO MAKE SHORT REPORT
 9405  043362  104031                          ERROR   31              ;MAKE FULL REPORT
 9406  043364  000137  043624                  JMP     37$
 9407  043370  104055                  32$:    ERROR   55              ;ABREVIATED HVRC ERROR RPORT
 9408  043372  000137  042150                  JMP     ER2ENT          ;GO REPORT 2ND LEVEL ERROR
 9409  043376  032704  000200          23$:    BIT     #BSE,R4         ;TEST FOR BAD SECTOR ERROR
 9410  043402  001430                          BEQ     33$             ;NO - SKIP
 9411  043404  052737  000002  005504          BIS     #BSERR,RECODE   ;SET ERROR FLAG
 9412  043412  016500  000030                  MOV     P.DCYL(R5),R0   ;GET CYL NO.
 9413  043416  116501  000027                  MOVB    P.DTS+1(R5),R1  ;GET TRACK NO.
 9414  043422  042701  177774                  BIC     #177774,R1      ;CLEAR ALL BITS EXCEPT TRACK
 9415  043426  016502  000026                  MOV     P.DTS(R5),R2    ;GET SECTOR IN ERROR
 9416  043432  042702  177740                  BIC     #177740,R2      ;CLEAR ALL BITS EXCEPT SECTOR
 9417  043436  004737  040146                  JSR     PC,BDSRCK       ;GO SEE IF THIS SECTOR LISTED
 9418  043442  032737  001000  005504          BIT     #BADSEC,RECODE  ;TEST RESULT
 9419  043450  001065                          BNE     37$             ;YES - EXIT, NO ERROR OR REPORT
 9420  043452  104104                          ERROR   104             ;ELSE REPORT BAD BSE
 9421  043454  042737  000002  005504          BIC     #BSERR,RECODE   ;RESET BSE ERROR FLAG
 9422  043462  000460                          BR      37$             ;GO EXIT
 9423  043464  032704  100000          33$:    BIT     #DCK,R4         ;TEST IF DATA CHECK
 9424  043470  001435                          BEQ     36$
 9425  043472  052737  000020  005504          BIS     #DCKERR,RECODE  ;SET DATA CHECK ERROR IN RECOVERY CODE
 9426  043500  032704  000100                  BIT     #ECH,R4         ;TEST IF ECC IS HARD. IF
```

```
 9427  043504  001406                          BEQ     34$             ;YES SET UNCORRECTABLE IN
 9428  043506  052737  000040  005504          BIS     #ECCNC,RFCODE   ;RECOVERY FLAG AND A 0 IN
 9429  043514  005037  001206                  CLR     $REG12          ;REG12 TO INDICATE UNCORRECTABLE,
 9430  043520  000403                          BR      35$             ;A 1 IN REG 12 FOR CORRECTABLE
 9431  043522  012737  000001  001200  34$:    MOV     #1,$REG12
 9432  043530  105737  003121          35$:    TSTB    DERCNT          ;TEST IF FIRST ERROR
 9433  043534  001033                          BNE     37$             ;NO SKIP REPORT
 9434  043536  004737  044632                  JSR     PC,GTPKAD       ;GO GET PACK ADDRESS OF ERROR
 9435  043542  016537  000060  001202          MOV     P.EPOS(R5),$REG10 ;STORE ECC POSITION &
 9436  043550  016537  000062  001204          MOV     P.EPAT(R5),$REG11 ;PATTERN
 9437  043556  104032                          ERROR   32              ;REPORT DCK ERROR
 9438  043560  000137  043624                  JMP     37$
 9439  043564  032765  040000  000020  36$:    BIT     #WCE,P.CS2(R5)  ;TEST WRITE CHECK ERROR
 9440  043572  001414                          BEQ     37$
 9441  043574  042737  000200  005504          BIC     #ABORT,RECODE   ;CLEAR ABORT & SET WRITE
 9442  043602  052737  000100  005504          BIS     #WCERR,RECODE   ;CHECK ERROR IN RECODE
 9443  043610  105737  003121                  TSTB    DERCNT          ;TEST IF FIRST ERROR
 9444  043614  001003                          BNE     37$             ;NO - SKIP
 9445  043616  004737  044632                  JSR     PC,GTPKAD       ;GO GET ADDRESS OF ERROR
 9446  043622  104033                          ERROR   33              ;REPORT WCE
 9447
 9448  043624  032765  000020  000014  37$:    BIT     #DRVHRD,P.PRST(R5) ;TEST HARD ERROR
 9449  043632  001404                          BEQ     43$
 9450  043634  104036                          ERROR   36
 9451  043636  052737  000200  005504          BIS     #ABORT,RECODE
 9452
 9453  043644  032765  000040  000014  43$:    BIT     #DRVDSC,P.PRST(R5) ;TEST STATUS CHANGE NOT CLEARED
 9454  043652  001404                          BEQ     44$
 9455  043654  104037                          ERROR   37
 9456  043656  052737  000200  005504          BIS     #ABORT,RECODE
 9457
 9458  043664  032765  004000  000014  44$:    BIT     #NODSC,P.PRST(R5) ;IFST ATTENTION BUT NO FAULT OR DSC
 9459  043672  001404                          BEQ     46$
 9460  043674  104040                          ERROR   40
 9461  043676  052737  000200  005504          BIS     #ABORT,RECODE
 9462
 9463  043704  032765  000010  000014  46$:    BIT     #UEXATT,P.PRST(R5) ;TEST UNEXPECTED ATTENTION
 9464  043712  001404                          BEQ     ALLTRM
 9465  043714  104042                          ERROR   42
 9466  043716  052737  000200  005504          BIS     #ABORT,RECODE
 9467
 9468
 9469                                  ;ALL ERRORS MUST EXIT THROUGH THIS POINT
 9470
 9471  043724  012705  002630          ALLTRM: MOV     #PARM0,R5       ;RESTORE PARAMETER BLOCK SELECTION
 9472  043730  012737  042130  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE INTERRUPT VECTORS FOR RETRY
 9473  043736  012737  040652  003044          MOV     #ERRFRE,A.NORM  ;DRIVER OPERATIONS, IF ANY
 9474  043744  032737  000200  005504          BIT     #ABORT,RECODE   ;IF ABORT IS NOT SET AND
 9475  043752  001043                          BNE     48$             ;THE DRIVE IS READY (HAS NOT
 9476                                                                  ;CYCLED DOWN)
 9477  043754  013702  003036                  MOV     RKBAS,R2        ;GET BASE ADDRESS
 9478  043760  032762  000200  000012          BIT     #RDY,RKDS(R2)   ;TEST IF DRIVE READY SET
 9479  043766  001004                          BNE     47$             ;RECALIBRATE REQUIRED BIT IS SET
 9480  043770  052737  000200  005504          BIS     #ABORT,RECODE   ;ELSE ABORT WITH ABORT MESSAGE
 9481  043776  000431                          BR      48$
 9482  044000  032737  004000  005504  47$:    BIT     #RCLREQ,RECODE  ;IF RECALIBRATE IS REQUIRED
```

```
 9483  044006  001443                          BEQ     BGNRTY          ;FOR RETRY SET UP PARAM
 9484  044010  112737  000113  000001          MOVB    #RECAL,P.CMND   ;BLOCK TO DO IT.
 9485  044016  012737  044270  003046          MOV     #RETANL,A.ABNL
 9486  044024  004737  040252                  JSR     PC,DRVCAL
 9487  044030  012737  042130  003046          MOV     #ERRHDL,A.ABNL  ;RESTORE ERROR RETURN
 9488  044036  032737  000400  005504          BIT     #LEV2ER,RECODE  ;IF AN ERROR OCCURRED IN THE
 9489  044044  001424                          BEQ     BGNRTY          ;RECAL ATTEMP SET ABORT,
 9490  044046  052737  000200  005504          BIS     #ABORT,RECODE   ;PRINT THE RECAL ERROR MESSAGE, AND
 9491  044054  104060                          ERROR   60              ;GO REPORT DETAILS
 9492  044056  000137  042150                  JMP     ER2ENT
 9493  044062  104061                  48$:    ERROR   61              ;REPORT ABORT-RETRY FAILED
 9494  044064  105037  003130                  CLRB    DRVERS          ;DROP DRIVE & GO TO NEW DRIVE
 9495  044070  105037  003126                  CLRB    ERRCNT
 9496  044074  000137  016754                  JMP     NEWDRV
 9497
 9498                                  ;THE PROGRAM WILL HALT HERE IF THE ABORT FLAG HAS BEEN SET OR
 9499                                  ;IF THE DRIVE READY BIT IS RESET.
 9500
 9501  044100  000005                  HLTPRG: RESET                   ;DISABLE ALL DEVICES
 9502  044102  000000                          HALT                    ;HALT THE CPU
 9503  044104  105037  003126                  CLRB    ERRCNT          ;CLEAR ERROR COUNT
 9504  044110  000005                          RESET                   ;RESET ALL DEVICES
 9505  044112  000137  012534                  JMP     CMSTRT
 9506
 9507
 9508                                  ;THE FOLLOWING CODE WILL DETERMINE IF AND DATA TYPE ERROR
 9509                                  ;HAS OCCURRED. IF YES, THE RETRY IS NOT DONE HERE BUT RETURNS TO
 9510                                  ; THE INITIATING ROUTINE FOR RETRY.  ANY OTHER ERROR IS TO BE
 9511                                  ;RETRIED HERE. IF RETRY IS UNSUCCESSFUL AFTER 4 ATTEMPTS, THE ABORT
 9512                                  ;FLAG IS SET AND PROGRAM HALTS.
 9513
 9514  044116  032737  000136  005504  BGNRTY: BIT     #BSERR!HVRCER!OPIERR!DCKERR!WCERR,RECODE ;TEST IF ANY DATA ERROR
 9515  044124  001404                          BEQ     3$
 9516  044126  052737  100000  005504  9$:     BIS     #ANYDER,RECODE
 9517  044134  000453                          BR      2$
 9518  044136                          3$:
 9519  044136  105737  003143                  TSTB    NORTRY          ;SEE IF "NO-RETRY" FLAG SET
 9520  044142  001371                          BNE     9$              ;BR IF YES
 9521  044144  032737  001000  005504          BIT     #BADSEC,RECODE  ;TEST IF BAD SECTOR FLAG SET
 9522  044152  001044                          BNE     2$              ;IF YES-EXIT TO CALLER
 9523  044154  123737  003126  003127          CMPB    ERRCNT,ERRLMT   ;BEGIN RETRY IF ERROR COUNT HAS
 9524  044162  001012                          BNE     1$              ;NOT BEEN EXCEEDED
 9525  044164  005037  001174                  CLR     $REG5
 9526  044170  113737  003126  001174          MOVB    ERRCNT,$REG5    ;GET ERROR RETRY COUNT
 9527  044176  104102                          ERROR   102             ;REPORT RETRY UNSUCCESSFUL
 9528  044200  052737  000200  005504          BIS     #ABORT,RECODE   ;SET ABORT & QUIT
 9529  044206  000734                          BR      HLTPRG
 9530  044210  013702  003036          1$:     MOV     RKBAS,R2        ;GET RK BASE ADDRESS
 9531  044214  112765  000177  000001          MOVB    #SUBCLR,P.CMND(R5)  ;SET UP TO CLEAR SUBSYSTEM
 9532  044222  004737  040252                  JSR     PC,DRVCAL       ;GO DO IT
 9533  044226  012700  005246                  MOV     #COMSTR,R0      ;GO AND REESTABLISH THE COMMAND
 9534  044232  012025                          MOV     (R0)+,(R5)+     ;AS IT WAS ENTERED INTO THE
 9535  044234  012025                          MOV     (R0)+,(R5)+     ;PARAMETER BLOCK
 9536  044236  012025                          MOV     (R0)+,(R5)+
 9537  044240  012025                          MOV     (R0)+,(R5)+
 9538  044242  012025                          MOV     (R0)+,(R5)+
```

```
9539  044244  012025                          MOV     (R0)+,(R5)+
9540  044246  012705  002630                  MOV     #PARM0,R5
9541  044252  012737  000000  177776          MOV     #PR0,@#PSW      ;LOWER PRIORITY TO ALLOW INTERRUPT
9542  044260  004737  040252                  JSR     PC,DRVCAL       ;CALL DRIVER
9543  044264  104410                  2$:     RESREG                  ;IF RETURN GETS HERE, NO ERROR
9544  044266  000207                          RTS     PC              ;OCCURRED, RECOVERY WAS SUCCESSFUL
9545
9546
9547  044270  152737  000377  003123  RETANL: BISB    #377,DONE       ;SET DONE
9548  044276  052737  000400  005504          BIS     #LEV2ER,RECODE  ;SET LEVEL TWO ERROR
9549  044304  000207                          RTS     PC
9550  044306  152737  000377  003123  RETNML: BISB    #377,DONE       ;SET DONE
9551  044314  000207                          RTS     PC
9552
9553
9554                                  ;;*****************************************************************
9555                                  .SBTTL  TIME OUT PROCESSOR ROUTINE
9556                                  ;*THIS ROUTINE SUPPORTS THE ERROR HANDLER BY PROCESSING TIME OUT STATUS
9557                                  ;*GATHERING DUTIES.
9558                                  ;;*****************************************************************
9559  044316                          TOPROC:
9560  044316  104407                          SAVREG
9561  044320  013702  003036                  MOV     RKBAS,R2
9562  044324  012701  001174                  MOV     #$REG5,R1       ;SET UP R1 FOR RK REGISTER STORAGE
9563  044330  016221  000000                  MOV     RKCS1(R2),(R1)+ ;STORE ALL VALID RK611
9564  044334  016221  000010                  MOV     RKCS2(R2),(R1)+ ;REGISTERS
9565  044340  016221  000020                  MOV     RKDC(R2),(R1)+
9566  044344  016221  000006                  MOV     RKDA(R2),(R1)+
9567  044350  016221  000002                  MOV     RKWC(R2),(R1)+
9568  044354  016221  000004                  MOV     RKBA(R2),(R1)+
9569  044360  016221  000016                  MOV     RKASOF(R2),(R1)+
9570  044364  016221  000012                  MOV     RKDS(R2),(R1)+
9571  044370  016221  000014                  MOV     RKER(R2),(R1)+
9572  044374  005000                          CLR     R0              ;THIS CODE WILL ATTEMPT TO
9573                                                                  ;RETRIEVE THE STATUS FROM THE
9574                                                                  ;DRIVE.
9575  044376  012705  002714                  MOV     #PARM1,R5       ;SET UP TO USE PARM1
9576  044402  112765  000141  000001          MOVB    #RDSTAT,P.CMND(R5) ;DO READ DRIVE STATUS COMMAND
9577  044410  004737  040252                  JSR     PC,DRVCAL       ;CALL DRIVER
9578  044414  032765  000100  000014          BIT     #CMDTO,P.PRST(R5) ;TEST FOR TIMEOUT
9579  044422  001403                          BEQ     1$              ;NO - SKIP
9580  044424  052737  002000  005504          BIS     #TWOTOS,RECODE  ;SET TWO TIMEOUTS FLAG
9581  044432  062705  000040          1$:     ADD     #P.A00,R5       ;BUMP R5 TO POINT TO DRIVE STATUS
9582  044436  012521                          MOV     (R5)+,(R1)+     ;MOVE ALL THE DRIVE STATUS INTO THE
9583  044440  012521                          MOV     (R5)+,(R1)+     ;TEMP REGS FOR REPORTING.
9584  044442  012521                          MOV     (R5)+,(R1)+
9585  044444  012521                          MOV     (R5)+,(R1)+
9586  044446  012521                          MOV     (R5)+,(R1)+
9587  044450  012521                          MOV     (R5)+,(R1)+
9588  044452  012521                          MOV     (R5)+,(R1)+
9589  044454  012521                          MOV     (R5)+,(R1)+
9590
9591  044456  012705  002630                  MOV     #PARM0,R5       ;RESTORE PARM 0
9592  044462  104410                          RESREG
9593  044464  000207                          RTS     PC
9594
```

```
                                        ;;*****************************************************************
                                        .SBTTL  READ HEADER 0 ROUTINE
                                        ;;*****************************************************************
044466                                  RDHD0:
044466  104407                                  SAVREG
044470  016501  000026                          MOV     P.DTS(R5),R1    ;STORE TRACK AND SECTOR
044474  016500  000052                          MOV     P.B10(R5),R0    ;GET THE CYLINDER ADRS
044500  042700  160017                          BIC     #160017,R0      ;FROM THE DRIVE STATUS. CLEAR
044504  006200                                  ASR     R0              ;OFF UNUSED BITS AND POSITION
044506  006200                                  ASR     R0              ;FOR USE AS THE DESIRED
044510  006200                                  ASR     R0              ;CYLINDER IN THE READ
044512  006200                                  ASR     R0              ;HEADER COMMAND.
044514  012705  002714                          MOV     #PARM1,R5       ;SET UP TO USE P.B  1
044520  010165  000004                          MOV     R1,P.SECT(R5)   ;INSERT TRK,SECT FOR READ HDR
044524  010065  000002                          MOV     R0,P.CYLN(R5)   ;SET CYL NO.
044530  112765  000177  000001                  MOVB    #SUBCLR,P.CMND(R5) ;SET S.S. CLEAR CMND
044536  012737  000000  177776                  MOV     #PR0,@#PSW      ;ALLOW INTERRUPTS
044544  004737  040252                          JSR     PC,DRVCAL       ;CLEAR THE S.S.
044550  112765  000125  000001                  MOVB    #RDHEAD,P.CMND(R5) ;SET READ HEADER COMMAND
044556  004737  040252                          JSR     PC,DRVCAL       ;DO READ HEADER
044562  013712  017716                          MOV     HOLD3,(R2)      ;ISSUE RD HDR WITH FMT BIT = 0
044566  032712  000200          4$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
044572  001775                          BEQ     4$              ;BR IF NOT RDY YET
044574  013712  017720                          MOV     HOLD4,(R2)      ;ISSUE RD HDR WITH FMT BIT = 1
044600  032712  000200          6$:     BIT     #BIT7,(R2)      ;CHECK FOR C. READY IN CS1
044604  001775                          BEQ     6$              ;BR IF NOT RDY YET
044606  142765  000020  000007          BICB    #B.CFMT,P.CS1H(R5) ;CLEAR THE FORMAT BIT
044614  153765  003125  000007          BISB    FORMAT,P.CS1H(R5) ;RESTORE TYPE AND FMT IN USE
044622  012705  002630                  MOV     #PARM0,R5       ;RESTORE P.B. 0 ADDRESS
044626  104410                          RESREG                  ;RESTORE R0-R5
044630  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        .SBTTL  GET PACK ADDRESS ROUTINE
                                        ;;*****************************************************************
044632  016537  000030  001174  GTPKAD: MOV     P.DCYL(R5),$REG5 ;GET CYLINDER NUMBER
044640  005037  001176                  CLR     $REG6           ;CLEAR REGISTERS FOR
044644  005037  001200                  CLR     $REG7           ;TRACK & SECTOR STORAGE
044650  116537  000026  001200          MOVB    P.DTS(R5),$REG7 ;STORE THE TRACK AND SECTOR
044656  116537  000027  001176          MOVB    P.DTS+1(R5),$REG6
044664  005737  001200                  TST     $REG7            ;ADJUST THE ADDRESS CONTAINED IN
                                                                ;THE RK REGISTERS FOR THE AUTOMATIC
044670  001403                          BEQ     1$              ;INCREMENT
044672  005337  001200                  DEC     $REG7
044676  000426                          BR      5$
044700  032765  010000  000016  1$:     BIT     #CFMT,P.CS1(R5)
044706  001404                          BEQ     2$
044710  012737  000023  001200          MOV     #19.,$REG7
044716  000403                          BR      3$
044720  012737  000025  001200  2$:     MOV     #21.,$REG7
044726  005737  001176          3$:     TST     $REG6
044732  001403                          BEQ     4$
```

```
 9651  044734  005337  001176                 DEC     $REG6
 9652  044740  000405                         BR      5$
 9653  044742  012737  000002  001176  4$:    MOV     #2,$REG6
 9654  044750  005337  001174                 DEC     $REG5
 9655  044754  000207                  5$:    RTS     PC
 9656
 9657
 9658
 9659                                 ;;*****************************************************************
 9660                                 .SBTTL  BUILD EXPECTED HEADER
 9661                                 ;*USES DESIRED CYLINDER, TRACK AND SECTOR REGISTERS TO DETERMINE
 9662                                 ;*WHICH HEADER WAS EXPECTED. LOADS EXPECTED VALUES IN $REG5, 6, AND
 9663                                 ;*7 FOR REPORTING.
 9664                                 ;;*****************************************************************
 9665  044756  104407                 BLDEXH: SAVREG
 9666  044760  016537  000030  001174         MOV     P.DCYL(R5),$REG5 ;CONSTRUCT EXPECTED HDR
 9667  044766  016501  000026                 MOV     P.DTS(R5),R1 ;DESIRED CYLINDER & DESIRED TRACK
 9668  044772  042701  174377                 BIC     #174377,R1      ;CLEAR ALL BUT TRACK BITS
 9669  044776  006201                         ASR     R1              ;AND SECTOR. SHIFT THE TRACK
 9670  045000  006201                         ASR     R1              ;OVER TO CONFORM TO HEADER FORMAT
 9671  045002  006201                         ASR     R1              ;CHECK THE FORMAT BIT AND
 9672                                                                 ;IF SET, SET THE HEADER FORMAT
 9673  045004  016537  000026  001176         MOV     P.DTS(R5),$REG6 ;BIT.
 9674  045012  042737  177740  001176         BIC     #177740,$REG6   ;CLEAR ALL BUT SECTOR
 9675  045020  060137  001176                 ADD     R1,$REG6        ;ADD TRACK AND SECTOR TOGETHER
 9676  045024  052737  140000  001176         BIS     #140000,$REG6   ;INSERT BSE BITS
 9677  045032  032765  010000  000016         BIT     #CFMT,P.CS1(R5)
 9678  045040  001403                         BEQ     23$
 9679  045042  052737  001000  001176         BIS     #1000,$REG6
 9680  045050  013737  001174  001200  23$:    MOV     $REG5,$REG7     ;COMPUTE THE HEADER VRC
 9681  045056  013701  001176                 MOV     $REG6,R1
 9682  045062  043737  001176  001200         BIC     $REG6,$REG7
 9683  045070  043701  001174                 BIC     $REG5,R1
 9684  045074  050137  001200                 BIS     R1,$REG7
 9685  045100  104410                         RESREG
 9686  045102  000207                         RTS     PC
 9687
```

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

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

                                .SBTTL   *WATCH-DOG TIMER

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

045104  010546                  W.WTCH: MOV     R5,-(SP)        ;SAVE R5 ON THE STACK
045106  010446                          MOV     R4,-(SP)        ;SAVE R4 ON THE STACK
045110  010346                          MOV     R3,-(SP)        ;SAVE R3 ON THE STACK
045112  010246                          MOV     R2,-(SP)        ;SAVE R2 ON STACK
045114  013746  177776                  MOV     PS,-(SP)        ;SAVE PROGRAM STATUS WORD ON STACK
045120  005337  003056                  DEC     W.MTIM          ;DECREMENT MILLISECOND TIMER
045124  001034                          BNE     20$             ;IF NOT ZERO RETURN
045126  013737  003060  003056          MOV     W.MILI,W.MTIM   ;REINITIALIZE MILLISECOND TIMER
045134  105737  003100                  TSTB    W.TIME          ;CHECK IF DRIVE IS BEING TIMED
045140  001426                          BEQ     20$             ;NO, RETURN
045142  013737  003042  177776          MOV     RKPRI,PS        ;LOCK OUT RK06-RK07 INTERRUPTS
045150  013702  003036                  MOV     RKBAS,R2        ;LOAD BASE OF RK06-RK07 REGISTERS
045154  005337  003114                  DEC     W.DRV           ;DECREMENT COMMAND TIMER
045160  001016                          BNE     20$             ;RETURN IF NO TIME OUT
```

```
 9744  045162  105037  003100                  CLRB    W.TIME          ;RESET TIMING INDICATOR
 9745  045166  013705  003112                  MOV     PBLKT,R5        ;LOAD ADDRESS OF PARAMETER BLOCK
 9746                                                                  ;  TABLE FOR INDEXING
 9747  045172  052765  000100  000014          BIS     #CMDTO,P.PRST(R5) ;SET COMMAND TIME OUT
 9748  045200  020537  003054                  CMP     R5,O.WAIT       ;CHECK IF DRIVER IS WAITING FOR
 9749                                                                  ;  COMMAND COMPLETION
 9750  045204  001002                          BNE     5$              ;NO, DO NOT ALTER WAITING FOR
 9751                                                                  ;  COMMAND COMPLETION
 9752  045206  005037  003054                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9753  045212  004737  050466          5$:     JSR     PC,R.ABNL       ;BRANCH TO ERROR ROUTINE
 9754  045216  012637  177776          20$:    MOV     (SP)+,PS        ;RESTORE PSW
 9755  045222  012602                          MOV     (SP)+,R2        ;RESTORE R2
 9756  045224  012603                          MOV     (SP)+,R3        ;RESTORE R3
 9757  045226  012604                          MOV     (SP)+,R4        ;RESTORE R4
 9758  045230  012605                          MOV     (SP)+,R5        ;RESTORE R5
 9759  045232  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 ADNORMAL 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
                                ;*
                                ;;*****************************************************************

045234  010546                  I.INTR: MOV     R5,-(SP)          ;STORE R5 ON THE STACK
045236  010446                          MOV     R4,-(SP)          ;STORE R4 ON THE STACK
045240  010346                          MOV     R3,-(SP)          ;STORE R3 ON THE STACK
045242  010246                          MOV     R2,-(SP)          ;STORE R2 ON THE STACK
045244  010146                          MOV     R1,-(SP)          ;STORE R1 ON THE STACK
045246  010046                          MOV     R0,-(SP)          ;STORE R0 ON THE STACK
045250  013702  003036                  MOV     RKBAS,R2          ;LOAD R2 TO ADDRESS RK06 REGISTER
045254  016237  000010  003002          MOV     RKCS2(R2),T.CS2   ;STORE CS2
045262  032737  001000  003002          BIT     #MDS,T.CS2        ;CHECK IF MULTIPLE DRIVE SELECT
045270  001407                        BEQ     1$                ;NO, CONTINUE PROCESSING
045272  052737  100000  003052        BIS     #E.MDS,E.CONT     ;SET MULTIPLE DRIVE SELECT
045300  004737  050512                JSR     PC,R.CONT         ;REPORT ERROR
```

```
 9816  045304  000137  047464                    JMP     I.RTRN          ;RETURN
 9817
 9818  045310  105737  003074          1$:       TSTB    I.ISRL          ;CHECK IF INTERRUPT OR RELEASE
 9819  045314  001410                             BEQ     6$              ;NO, CHECK IF DRIVE AVAILABLE
 9820  045316  100403                             BMI     5$              ;CHECK IF RELEASE COMMAND
 9821  045320  105037  003074                    CLRB    I.ISRL          ;YES,CLEAR FLAG
 9822  045324  000473                             BR      I.IOO           ;CONTINUE PROCESSING INTERRUPT
 9823
 9824  045326  105037  003074          5$:       CLRB    I.ISRL          ;CLEAR FLAG
 9825  045332  000137  046446                    JMP     I.ATTN          ;GO PROCESS DRIVE ATTENTIONS
 9826
 9827  045336  032737  010400  003002  6$:       BIT     #NED!UFE,T.CS2  ;CHECK FOR NON-EXISTENT DRIVE OR
 9828                                                                    ;  UNIT FIELD ERROR
 9829  045344  001413                             BEQ     7$              ;NO, WAIT FOR DUAL ACCESS INTERRUPT
 9830  045346  013704  003002                    MOV     T.CS2,R4        ;LOAD R4 FOR DRIVE NUMBER
 9831  045352  042704  177770                    BIC     #^C<DRVMSK>,R4  ;KEEP DRIVE BITS
 9832  045356  013705  003112                    MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS
 9833  045362  016237  000000  003000            MOV     RKCS1(R2),T.CS1 ;LOAD TEMPORARY CS1 FOR STATUS REPORT
 9834  045370  000137  045656                    JMP     I.ERRC          ;REPORT ERROR
 9835
 9836  045374  016237  000012  003020  7$:       MOV     RKDS(R2),T.DS   ;STORE STATUS REGISTER FOR COMPARISON
 9837  045402  032737  000001  003020            BIT     #DRA,T.DS       ;CHECK IF DRIVE SEIZED BY OTHER
 9838                                                                    ;  PORT
 9839  045410  001041                             BNE     I.IOO           ;NO, CONTINUE PROCESSING INTERRUPT
 9840
 9841                                            ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
 9842  045412  032737  164000  003002            BIT     #DLT!WCE!UPE!NEM,T.CS2
 9843
 9844  045420  001007                             BNE     10$             ;INDICATE ERROR
 9845  045422  016237  000014  003016            MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
 9846
 9847                                  ;         CHECK FOR DATA TRANSFER ERROR TYPE ERROR
 9848  045430  032737  125700  003016            BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
 9849
 9850  045436  001407                             BEQ     11$             ;NO, WAIT FOR RELEASE OF RK06 DRIVE
 9851
 9852  045440  052737  000010  003052  10$:      BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
 9853  045446  004737  050512                    JSR     PC,R.CONT       ;REPORT ERROR
 9854  045452  000137  047464                    JMP     I.RTRN          ;RESTORE REGISTERS
 9855
 9856  045456  105037  003100          11$:      CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 9857  045462  005037  003114                    CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
 9858  045466  013705  003112                    MOV     PBLKT,R5        ;LOAD R5 WITH PARAMETER BLOCK
 9859                                                                    ;  ADDRESS
 9860  045472  052765  010000  000014            BIS     #DRVSZD,P.PRST(R5) ;SET DRIVE SEIZED IN THE
 9861                                                                    ;  PROGRAM DRIVE STATUS REGISTER
 9862  045500  005037  003054                    CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9863  045504  004737  050466                    JSR     PC,R.ABNL       ;INDICATE ABNORMAL TERMINATION
 9864  045510  000137  047464                    JMP     I.RTRN          ;GO RESTORE REGISTERS
 9865
 9866  045514  013705  003054          I.IOO:    MOV     O.WAIT,R5       ;LOAD PARAMETER BLOCK ADDRESS INTO R5
 9867  045520  001002                             BNE     2$              ;IS COMMAND WAITING PROCESSING
 9868                                                                    ;  YES, DO PROCESSING
 9869  045522  000137  046446                    JMP     I.ATTN          ;NO, PROCESS ATTENTION
 9870
 9871  045526  013704  003002          2$:       MOV     T.CS2,R4        ;STORE RKCS2 FOR DRIVE NUMBER
```

```
 9872  045532  042704  177770                    BIC     #^C<DRVMSK>,R4  ;MASK OUT UNNECESSARY BITS
 9873
 9874
 9875  045536  126504  000000                    CMPB    P.DRVN(R5),R4   ;CHECK IF DRIVE NUMBER IS EXPECTED
 9876  045542  001401                            BEQ     3$              ;YES, CONTINUE
 9877  045544  000000                            HALT                    ;NO, DRIVER ERROR
 9878  045546  122765  000164  000001  3$:       CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 9879  045554  001002                            BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
 9880  045556  000137  046114                    JMP     I.HDAL          ;GO EXECUTE SPECIAL HEADER SEQUENCE
 9881
 9882  045562  005037  003054          10$:      CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND COMPLETION
 9883  045566  005037  003114                    CLR     W.DRV           ;CLEAR WATCH-DOG TIME
 9884  045572  105037  003100                    CLRB    W.TIME          ;RESET TIMING ON THIS DRIVE
 9885  045576  016237  000000  003000            MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
 9886  045604  032737  100000  003000            BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
 9887  045612  001021                            BNE     I.ERRC          ;YES, PROCESS ERROR
 9888  045614  016237  000016  003014            MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
 9889  045622  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
 9890  045630  001004                            BNE     15$             ;YES, REPORT ERROR
 9891  045632  004737  050500                    JSR     PC,R.NORM       ;INDICATE NORMAL RETURN
 9892  045636  000137  047464                    JMP     I.RTRN          ;RESTORE REGISTERS
 9893
 9894  045642  052765  000010  000014  15$:      BIS     #UEXATT,P.PRST(R5) ;SET UNEXPECTED ATTENTION
 9895
 9896  045650  004737  050134          I.ERRA:   JSR     PC,I.CSTS       ;STORE CONTROLLER STATUS
 9897  045654  000405                            BR      I.ERR           ;STORE PATTERN AND POSITION INFORMATION
 9898
 9899  045656  013765  003000  000016  I.ERRC:   MOV     T.CS1,P.CS1(R5) ;GET ERROR RKCS1
 9900  045664  004737  050156                    JSR     PC,I.CST1       ;GET REST OF CONTROLLER STATUS
 9901  045670  016265  000032  000062  I.ERR:    MOV     RKECPT(R2),P.EPAT(R5) ;STORE ECC PATTERN
 9902  045676  016265  000030  000060            MOV     RKECPS(R2),P.EPOS(R5) ;STORE ECC POSITION
 9903  045704  004037  047502                    JSR     R0,I.CCLR       ;CLEAR CONTROLLER
 9904  045710  047464                            I.RTRN                  ;ERROR RETURN
 9905  045712  032765  010400  000020            BIT     #NED!UFE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE OR
 9906                                                                    ;  UNIT FIELD ERROR
 9907  045720  001046                            BNE     5$              ;YES, REPORT ERROR
 9908  045722  004037  050240                    JSR     R0,I.STAT       ;GATHER DRIVE STATUS
 9909  045726  047464                            I.RTRN                  ;ERROR RETURN
 9910  045730  112737  000005  003000            MOVB    #DR.CLR,T.CS1   ;LOAD COMMAND
 9911  045736  004037  047564                    JSR     R0,I.ISSU       ;ISSUE DRIVE CLEAR
 9912  045742  047464                            I.RTRN                  ;ERROR RETURN
 9913  045744  133737  003101  003015            BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
 9914  045752  001407                            BEQ     2$              ;NO, INDICATE DRIVE ERROR
 9915  045754  052737  000020  003052            BIS     #E.CLAT,E.CONT  ;SET ATTENTION DID NOT RESET
 9916                                                                    ;  WITH CLEAR
 9917  045762  004737  050512                    JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
 9918  045766  000137  047464                    JMP     I.RTRN          ;GO RESTORE REGISTERS
 9919
 9920  045772  032737  040000  003024  2$:       BIT     #S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
 9921  046000  001403                            BEQ     3$              ;YES, CHECK FAULT
 9922  046002  052765  000040  000014            BIS     #DRVDSC,P.PRST(R5) ;SET DSC DID NOT CLEAR
 9923  046010  032737  001000  003026  3$:       BIT     #S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
 9924  046016  001407                            BEQ     5$              ;NO, INDICATE ABNORMAL TERMINATION
 9925  046020  052737  002000  003052            BIS     #E.DPAR,E.CONT  ;SET DRIVE PARITY ERROR
 9926  046026  004737  050512                    JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
 9927  046032  000137  047464                    JMP     I.RTRN          ;RETURN
```

```
9928
9929   046036  032765  000020  000014  5$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
9930   046044  001017                          BNE     10$             ;YES, GO REPORT ERROR
9931   046046  032737  020000  003024          BIT     #S.PIP,T.MR2    ;CHECK IF DRIVE IS CYCLING DOWN
9932   046054  001413                          BEQ     10$             ;NO, REPORT ERROR
9933   046056  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
9934   046064  113737  003101  003100          MOVB    INTMSK,W.TIME   ;SET UP 8 SECONDS FOR DRIVE TO CYCLE UP
9935   046072  013737  003064  003114          MOV     W.8SEC,W.DRV
9936   046100  000137  047464                  JMP     I.RTRN          ;GO RESTORE REGISTERS
9937
9938   046104  004737  050466          10$:    JSR     PC,R.ABNL       ;GO REPORT ERROR
9939   046110  000137  047464                  JMP     I.RTRN          ;GO RESTORE REGISTERS
9940
9941                                  .SBTTL   *READ ALL HEADERS INTERRUPT SEQUENCE
9942
9943   046114  016237  000000  003000  I.HDAL: MOV     RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLEP
9944                                                                   ;  ERROR
9945   046122  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR
9946   046130  001422                          BEQ     5$              ;NO, CHECK FOR ATTENTION
9947
9948   046132  005037  003054                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETE
9949   046136  105037  003100                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
9950   046142  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
9951   046146  013765  003000  000016          MOV     T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
9952   046154  004737  050156                  JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
9953   046160  004037  047502                  JSR     R0,I.CCLR       ;CLEAR CONTROLLER
9954   046164  047464                          I.RTRN                  ;ERROR RETURN
9955   046166  004737  050466                  JSR     PC,R.ABNL       ;INDICATE ERROR RETURN
9956   046172  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
9957
9958   046176  016237  000016  003014  5$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
9959   046204  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
9960   046212  001410                          BEQ     7$              ;NO, CHECK IF READ ALL HEADERS
9961   046214  005037  003054                  CLR     O.WAIT          ;CLEAR WAITING FOR COMMAND COMPLETION
9962   046220  105037  003100                  CLRB    W.TIME          ;RESET TIMING ON DRIVE
9963   046224  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT
9964   046230  000137  045650                  JMP     I.ERRA          ;GO REPORT ERROR
9965
9966   046234  013701  003070          7$:     MOV     HDR.AD,R1       ;GET MAIN MEMORY ADDRESS
9967   046240  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET FIRST WORD OF HEADER
9968   046244  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET SECOND WORD OF HEADER
9969   046250  016221  000024                  MOV     RKDB(R2),(R1)+  ;GET THIRD WORD OF HEADER
9970   046254  010137  003070                  MOV     R1,HDR.AD       ;STORE ADDRESS FOR NEXT HEADER
9971   046260  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
9972   046266  032737  100000  003002          BIT     #DLT,T.CS2      ;CHECK FOR DATA LATE
9973   046274  001055                          BNE     35$             ;YES, REPORT ERROR
9974   046276  005337  003072                  DEC     HDR.CT          ;DECREMENT NUMBER OF HEADER YET TO READ
9975   046302  001026                          BNE     25$             ;IF NON-ZERO, GO ISSUE NEXT READ HEADER
9976   046304  005037  003054                  CLR     O.WAIT          ;CLEAR DRIVER WAITING FOR COMMAND COMPLETION
9977   046310  005037  003114                  CLR     W.DRV           ;CLEAR TIME OUT COUNT FOR THIS DR'VE
9978   046314  105037  003100                  CLRB    W.TIME          ;CLEAR WATCH DOG TIME ON THIS DRIVE
9979   046320  012762  000003  000026          MOV     #3,RKMR1(R2)    ;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
9980   046326  112737  000001  003000          MOVB    #DR.SEL,T.CS1   ;LOAD SELECT COMMAND
9981   046334  004037  047564                  JSR     R0,I.ISSU       ;GET SECTOR COUNT
9982   046340  047464                          I.RTRN                  ;ERROR RETURN
9983   046342  013765  003026  000056          MOV     T.MR3,P.B11(R5) ;LOAD SECTOR COUNT
```

```
 9984  046350  004737  050500                  JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
 9985  046354  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
 9986
 9987  046360  016562  000002  000020  25$:    MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
 9988  046366  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
 9989  046374  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 9990  046402  042765  165777  000016          BIC     #^C<CDT!CFMT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT AND
 9991                                                                  ;  DRIVE TYPE
 9992  046410  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5)  ;STORE COMMAND ISSUED
 9993  046416  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2)  ;ISSUE READ HEADER
 9994  046424  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
 9995
 9996  046430  052737  000400  003052  35$:    BIS     #E.DLT,E.CONT   ;SET DATA LATE WHILE UNLOADING HEADER
 9997  046436  004737  050512                  JSR     PC,R.CONT       ;REPORT ERROR
 9998  046442  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
 9999
10000                                  .SBTTL   *DRIVE ATTENTION SCANNER
10001
10002  046446  016237  000000  003000  I.ATTN: MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS
10003                                                                  ;  REGISTER 1 FOR COMPARISON
10004  046454  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCURRED
10005  046462  001441                          BEQ     5$              ;NO, CHECK IF ATTENTION
10006
10007                                          ;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
10008  046464  032737  164000  003002          BIT     #DLT!WCE!UPE!NEM,T.CS2
10009
10010  046472  001007                          BNE     1$              ;INDICATE ERROR
10011  046474  016237  000014  003016          MOV     RKER(R2),T.ER   ;STORE ERROR REGISTER
10012
10013                                  ;       CHECK FOR DATA TRANSFER ERROR TYPE
10014  046502  032737  125700  003016          BIT     #DCK!OPI!WLE!COE!HVRC!BSE!ECH,T.ER
10015
10016  046510  001407                          BEQ     2$              ;NO DATA TRANSFER ERROR
10017
10018  046512  052737  000010  003052  1$:     BIS     #E.UDAT,E.CONT  ;SET UNEXPECTED DATA TYPE ERROR
10019  046520  004737  050512                  JSR     PC,R.CONT       ;REPORT ERROR
10020  046524  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
10021
10022  046530  013704  003002          2$:     MOV     T.CS2,R4        ;SAVE CS2 FOR REGISTER NUMBER
10023  046534  042704  177770                  BIC     #^C<DRVMSK>,R4  ;STRIP OFF JUNK
10024  046540  105037  003100                  CLRB    W.TIME          ;CLEAR WATCH DOG TIMER
10025  046544  005037  003114                  CLR     W.DRV           ;RESET TIMER VALUE
10026  046550  013705  003112                  MOV     PBLKT,R5        ;STORE PARAMETER BLOCK ADDRESS IN R5
10027
10028                                  ;       CLEAR DRIVE POSITIONING AND DRIVE POSITIONED FOR DATA TRANSFER
10029                                  ;         IN PROGRAM DEVICE STATUS REGISTER
10030  046554  042765  000006  000014          BIC     #DRVPOS!DRVPDT,P.PRST(R5)
10031
10032  046562  000137  045656                  JMP     I.ERRC          ;GO REPORT ERROR
10033
10034  046566  032737  040000  003000  5$:     BIT     #DI,T.CS1       ;CHECK IF ANY DRIVE ATTENTION
10035  046574  001002                          BNE     6$              ;YES, PROCESS INTERRUPT
10036  046576  000137  047464                  JMP     I.RTRN          ;RESTORE REGISTERS
10037
10038  046602  016237  000016  003014  6$:     MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10039  046610  105737  003015                  TSTB    T.ASOF+1        ;CHECK IF ANY ATTENTIONS SET
```

```
10040  046614  001007                          BNE     7$                ;YES GO PROCESS INTERRUPT
10041  046616  052737  000002  003052          BIS     #E.NOAT,E.CONT    ;SET NO ATTENTION IN ATTENTION SUMMARY
10042  046624  004737  050512                  JSR     PC,R.CONT         ;GO REPORT ERROR
10043  046630  000137  047464                  JMP     I.RTRN            ;GO RESTORE REGISTERS
10044
10045  046634  133737  003101  003015  7$:     BITB    INTMSK,T.ASOF+1   ;CHECK IF DESIRED INTERRUPT
10046  046642  001007                          BNE     8$                ;YES, GO PROCESS IT
10047  046644  052737  000004  003052          BIS     #E.UATT,E.CONT    ;SET UNSOLICATED ATTENTION
10048  046652  004737  050512                  JSR     PC,R.CONT         ;GO REPORT ERROR
10049  046656  000137  047464                  JMP     I.RTRN            ;GO RESTORE REGISTERS
10050
10051  046662  013705  003112          8$:     MOV     PBLKT,R5          ;STORE PARAMETER BLOCK TABLE
10052  046666  116504  000000                  MOVB    P.DRVN(R5),R4     ;STORE DRIVE NUMBER
10053  046672  032765  020000  000014          BIT     #E.UNLD,P.PRST(R5)  ;CHECK IF DRIVE UNLOADING
10054  046700  001402                          BEQ     11$               ;NO, CONTINUE
10055  046702  000137  047364                  JMP     I.UNLD            ;SERVICE DRIVE IN POSITION AFTER ERROR
10056
10057  046706  042765  000002  000014  11$:    BIC     #DRVPOS,P.PRST(R5)  ;RESET DRIVE POSITIONING
10058  046714  005062  000026                  CLR     RKMR1(R2)         ;CLEAR MAINTENANCE REGISTER 1
10059  046720  112737  000001  003000          MOVB    #DR.SEL,T.CS1     ;LOAD COMMAND
10060  046726  004037  047564                  JSR     R0,I.ISSU         ;SELECT DRIVE WITH ATTENTION HIGH
10061  046732  047464                          I.RTRN                    ;ERROR RETURN
10062  046734  013765  003026  000042          MOV     T.MR3,P.B00(R5)   ;STORE STATUS BYTE 00 MESS B
10063  046742  032765  000200  000042          BIT     #S.FLT,P.B00(R5)  ;CHECK IF DRIVE FAULT
10064  046750  001401                          BEQ     12$               ;NO, CHECK FOR DRIVE STATUS CHANGE
10065  046752  000461                          BR      I.AERR            ;PROCESS ERROR
10066
10067  046754  013765  003024  000040  12$:    MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
10068  046762  032765  040000  000040          BIT     #S.DSC,P.A00(R5)  ;CHECK FOR DRIVE STATUS CHANGE
10069  046770  001004                          BNE     13$               ;YES, PROCESS DRIVE STATUS CHANGE
10070  046772  052765  004000  000014          BIS     #NODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
10071  047000  000446                          BR      I.AERR            ;PROCESS ERROR
10072
10073  047002  112737  000005  003000  13$:    MOVB    #DR.CLR,T.CS1     ;LOAD COMMAND
10074  047010  004037  047564                  JSR     R0,I.ISSU         ;CLEAR DRIVE STATUS CHANGE
10075  047014  047464                          I.RTRN                    ;ERROR RETUN
10076  047016  013765  003014  000032          MOV     T.ASOF,P.ASOF(R5) ;STORE ATTENTION SUMMARY
10077  047024  133765  003101  000033          BITB    INTMSK,P.ASOF+1(R5) ;CHECK IF ATTENTION RESET
10078  047032  001407                          BEQ     15$               ;YES, CONTINUE INTERRUPT PROCESSING
10079  047034  052737  000020  003052          BIS     #E.CLAT,E.CONT    ;SET ATTENTION DID NOT RESET
10080                                                                    ;  WITH DRIVE CLEAR
10081  047042  004737  050512                  JSR     PC,R.CONT         ;FLAG ERROR
10082  047046  000137  047464                  JMP     I.RTRN            ;RESTORE REGISTERS
10083
10084  047052  013765  003024  000040  15$:    MOV     T.MR2,P.A00(R5)   ;STORE MAINTENANCE REGISTER 2
10085  047060  032765  040000  000040          BIT     #S.DSC,P.A00(R5)  ;CHECK IF DRIVE STATUS CHANGE
10086                                                                    ;  RESET
10087  047066  001404                          BEQ     16$               ;YES, CONTINUE INTERRUPT PROCESSING
10088  047070  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
10089  047076  000407                          BR      I.AERR            ;GO PROCESS ERROR
10090
10091  047100  105C37  003100          16$:    CLRB    W.TIME            ;RESET TIMING ON THIS DRIVE
10092  047104  005037  003114                  CLR     W.DRV             ;CLEAR DRIVE TIMING COUNT
10093  047110  004737  050500                  JSR     PC,R.NORM         ;REPORT SUCCESSFUL COMMAND COMPLETETION
10094  047114  000563                          BR      I.RTRN            ;RESTORE REGISTERS
10095
```

```
                                .SBTTL   *ATTENTION ERROR HANDLER

047116  042765  000004  000014  I.AERR: BIC     #DRVPDT,P.PRST(R5) ;RESET POSITIONING IN PROGRESS BECAUSE
                                                                   ;  OF DATA TRANSFER
047124  105037  003100                  CLRB    W.TIME             ;CLEAR TIMING FOR THIS DRIVE
047130  005037  003114                  CLR     W.DRV              ;RESET WATCH-DOG TIME
047134  042765  177741  000016          BIC     #177741,P.CS1(R5) ;KEEP COMMAND ISSUED
047142  042737  000036  003000          BIC     #36,T.CS1          ;KEEP CURRENT CONTROLLER STATUS
047150  053765  003000  000016          BIS     T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
047156  013765  003002  000020          MOV     T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
047164  013765  003004  000022          MOV     T.WCR,P.WCR(R5)
047172  013765  003006  000024          MOV     T.BA,P.BAR(R5)
047200  013765  003010  000026          MOV     T.DA,P.DTS(R5)
047206  013765  003012  000030          MOV     T.DC,P.DCYL(R5)
047214  013765  003014  000032          MOV     T.ASOF,P.ASOF(R5)
047222  013765  003016  000034          MOV     T.ER,P.ER(R5)
047230  013765  003020  000036          MOV     T.DS,P.DS(R5)
047236  004037  050240                  JSR     R0,I.STAT          ;GATHER DRIVE STATUS
047242  047464                          I.RTRN                     ;ERROR RETURN
047244  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD COMMAND
047252  004037  047564                  JSR     R0,I.ISSU          ;CLEAR DRIVE ERRORS
047256  047464                          I.RTRN                     ;ERROR RETURN
047260  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF ATTENTION RESET
047266  001407                          BEQ     2$                 ;YES, FLAG DRIVE ERROR
047270  052737  000020  003052          BIS     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
047276  004737  050512                  JSR     PC,R.CONT          ;REPORT ERROR
047302  000137  047464                  JMP     I.RTRN             ;RESTORE REGISTERS

047306  032765  000020  000014  2$:     BIT     #DRVHRD,P.PRST(R5) ;CHECK IF AHARD DRIVE ERROR
047314  001017                          BNE     10$                ;YES, REPORT ERROR
047316  032737  020000  003024          BIT     #S.PIP,T.MR2       ;CHECK IF DRIVE IS UNLOADING
047324  001413                          BEQ     10$                ;NO, REPORT ERROR
047326  052765  020000  000014          BIS     #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO ERROR
047334  113737  003101  003100          MOVB    INTMSK,W.TIME      ;SET TIMING ON THIS DRIVE
047342  013737  003064  003114          MOV     W.8SEC,W.DRV       ;LOAD 8 SECONDS FOR CYCLE UP TIME
047350  000137  047464                  JMP     I.RTRN             ;RESTORE REGISTERS

047354  004737  050466          10$:    JSR     PC,R.ABNL          ;REPORT ERROR
047360  000137  047464                  JMP     I.RTRN             ;RESTORE REGISTERS

                                .SBTTL   *ERROR CAUSING DRIVE TO UNLOAD

047364  052765  020000  000014  I.UNLD: BIS     #E.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ERROR
047372  112737  000005  003000          MOVB    #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
047400  004037  047564                  JSR     R0,I.ISSU          ;GO ISSUE DRIVE CLEAR
047404  047464                          I.RTRN                     ;ERROR RETURN
047406  136437  003101  003015          BITB    INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
047414  001406                          BEQ     15$                ;YES, CONTINUE
047416  012737  000020  003052          MOV     #E.CLAT,E.CONT     ;SET ATTENTION DID NOT RESET
047424  004737  050512                  JSR     PC,R.CONT          ;REPORT ERROR
047430  000415                          BR      I.RTRN             ;RESTORE REGISTERS

047432  032737  040000  003024  15$:    BIT     #S.DSC,T.MR2       ;CHECK IF DRIVE STAUS CHANGE RESET
047440  001403                          BEQ     20$                ;YES, CONTINUE
047442  052765  000040  000014          BIS     #DRVDSC,P.PRST(R5) ;SET DRIVE STAUS CHANGE DID NOT CLEAR
047450  105037  003100          20$:    CLRB    W.TIME             ;RESET TIMING ON THIS DRIVE
```

```
10152  047454  005037  003114                  CLR     W.DRV           ;CLEAR TIME COUNT
10153  047460  004737  050466                  JSR     PC,R.ABNL       ;REPORT ERROR
10154
10155  047464  012600                I.RTRN:   MOV     (SP)+,R0        ;RESTORE R0
10156  047466  012601                          MOV     (SP)+,R1        ;RESTORE R1
10157  047470  012602                          MOV     (SP)+,R2        ;RESTORE R2
10158  047472  012603                          MOV     (SP)+,R3        ;RESTORE R3
10159  047474  012604                          MOV     (SP)+,R4        ;RESTORE R4
10160  047476  012605                          MOV     (SP)+,R5        ;RESTORE R5
10161  047500  000002                          RTI                     ;RETURN
10162
```

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

047502  012762  100000  000000  I.CCLR: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
047510  016237  000000  003000          MOV     RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
047516  032737  100000  003000          BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER CLEAR DID
                                                                ;  CLEAR ERROR
047524  001407                          BEQ     5$              ;YES, RETURN TO DRIVER PROCESSING
047526  052737  000001  003052          BIS     #E.CCLR,E.CONT  ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
047534  004737  050512                  JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
047540  011000                          MOV     (R0),R0         ;SET UP ERROR RETURN
047542  000200                          RTS     R0              ;RETURN

047544  012762  000100  000000  5$:     MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
047552  112737  177777  003074          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE ISSUED
047560  005720                          TST     (R0)+           ;ADJUST FOR NORMAL RETURN
047562  000200                          RTS     R0              ;RETURN
```

```
10198                                   .SBTTL   *COMMAND ISSUED BY DRIVER SERVICE ROUTINE
10199
10200                                   ;;*****************************************************************
10201                                   ;*
10202                                   ;*      THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
10203                                   ;*      AND CHECK IF A CONTROLLER ERROR OCCURRED.  IF A CONTROLLER
10204                                   ;*      ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
10205                                   ;*      CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
10206                                   ;*      ADDRESS IN A.CONT.
10207                                   ;*
10208                                   ;*      REGISTER        USE
10209                                   ;*      --------        ---
10210                                   ;*
10211                                   ;*      R2              ADDDRESS OF RK06 REGISTERS
10212                                   ;*      R5              ADDRESS OF PARAMETER BLOCK
10213                                   ;*
10214                                   ;*CALL  JSR     R0,I.ISSU
10215                                   ;*      <ADDRESS OF ERROR RETURN>
10216                                   ;*      RETURN
10217                                   ;*
10218                                   ;*      ROUTINES USED:
10219                                   ;*      --------------
10220                                   ;*
10221                                   ;*              I.CCLR
10222                                   ;*              I.STOR
10223                                   ;*
10224                                   ;;*****************************************************************
10225
10226  047564  013746  003000           I.ISSU: MOV     T.CS1,-(SP)     ;STORE COMMAND ISSUED
10227  047570  005037  003002                   CLR     T.CS2           ;CLEAR TEMPORARY CS2
10228  047574  116537  000000  003002           MOVB    P.DRVN(R5),T.CS2 ;LOAD IN DRIVE NUMBER
10229  047602  013762  003002  000010           MOV     T.CS2,RKCS2(R2) ;LOAD DRIVE NUMBER FOR COMMAND
10230  047610  116537  000007  003001           MOVB    P.CS1H(R5),T.CS1+1 ;STORE BITS 8-15 OF CS1
10231  047616  142737  177753  003001           BICB    #^C<B.CDT!B.CFMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
10232                                                                   ;  FORMAT AND DRIVE TYPE
10233  047624  013762  003000  000000           MOV     T.CS1,RKCS1(R2) ;ISSUE COMMAND
10234  047632  105762  000000           1$:     TSTB    RKCS1(R2)       ;WAIT FOR READY
10235  047636  100375                           BPL     1$
10236  047640  004737  050006                   JSR     PC,I.STOR       ;GO STORE REGISTERS
10237  047644  032737  100000  003000           BIT     #CERR,T.CS1     ;CHECK IF CONTROLLER ERROR OCCUERED
10238  047652  001437                           BEQ     5$              ;NO, RETURN
10239  047654  032737  001000  003002           BIT     #MDS,T.CS2      ;CHECK IF MULTIPLE DRIVE SELECT
10240  047662  001406                           BEQ     2$              ;NO, CHECK FOR OTHER CONTROLLER ERRORS
10241  047664  052737  100000  003052           BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT FLAG
10242  047672  004737  050512                   JSR     PC,R.CONT       ;REPORT CONTROLLER ERROR
10243  047676  000440                           BR      10$             ;RETURN
10244
10245                                           ;CHECK IF ANY CONTROLLER ERROR IS SET
10246  047700  032737  024000  003000   2$:     BIT     #CTO!SPAR,T.CS1
10247  047706  001027                           BNE     7$
10248  047710  032737  176400  003002           BIT     #UFE!PGE!NEM!NED!UPE!WCE!DLT,T.CS2
10249  047716  001023                           BNE     7$
10250  047720  032737  131761  003016           BIT     #ILC!DTYE!FMTE!ECH!BSE!HVRC!COE!DTE!OPI!DCK,T.ER
10251  047726  001017                           BNE     7$
10252
10253  047730  122716  000005                   CMPB    #DR.CLR,(SP)    ;CHECK IF CLEAR DRIVE
```

```
10254  047734  001003                      BNE     3$                ;NO, DO NOT SET DRIVE HARD ERROR
10255  047736  052765  000020  000014      BIS     #DRVHRD,P.PRST(R5) ;SET HARD DRIVE ERROR
10256  047744  004037  047502        3$:   JSR     R0,I.CCLR         ;GO ISSUE A CONTROLLER CLEAR
10257  047750  050000                      10$                       ;ERROR RETURN
10258  047752  012762  000100  000000  5$: MOV     #IE,RKCS1(R2)     ;SET INTERRUPT ENABLE
10259  047760  005726                      TST     (SP)+             ;ADJUST STACK
10260  047762  005720                      TST     (R0)+             ;ADJUST R0 FOR NORMAL RETURN
10261  047764  000200                      RTS     R0                ;RETURN
10262
10263  047766  052737  001000  003052  7$: BIS     #E.CERR,E.CONT    ;SET CONTROLLER ERROR DURING
10264                                              ;  DRIVER SERVICING
10265  047774  004737  050512              JSR     PC,R.CONT         ;REPORT ERROR
10266  050000  005726              10$:    TST     (SP)+             ;ADJUST STACK
10267  050002  011000                      MOV     (R0),R0           ;ADJUST R0 FOR ERROR RETURN
10268  050004  000200                      RTS     R0                ;RETURN
```

```
10269                                   .SBTTL   *STORE RK611 UNIBUS REGISTERS
10270
10271                                   ;;*****************************************************************
10272                                   ;*
10273                                   ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL
10274                                   ;*      RK611 REGISTER IN TEMPORARY LOCATIONS.
10275                                   ;*
10276                                   ;*CALL  JSR     PC,I.STOR
10277                                   ;*      RETURN
10278                                   ;*
10279                                   ;*      REGISTER        USE
10280                                   ;*      --------        ---
10281                                   ;*
10282                                   ;*      R2              ADDRESS OF RK611 REGISTERS
10283                                   ;*
10284                                   ;;*****************************************************************
10285
10286  050006  016237  000000  003000  I.STOR: MOV     RKCS1(R2),T.CS1 ;STORE ALL CONTROLLER REGISTERS
10287  050014  016237  000010  003002          MOV     RKCS2(R2),T.CS2 ;  EXCEPT DATA BUFFER
10288  050022  016237  000002  003004          MOV     RKWC(R2),T.WCR
10289  050030  016237  000004  003006          MOV     RKBA(R2),T.BA
10290  050036  016237  000006  003010          MOV     RKDA(R2),T.DA
10291  050044  016237  000012  003020          MOV     RKDS(R2),T.DS
10292  050052  016237  000014  003016          MOV     RKER(R2),T.ER
10293  050060  016237  000016  003014          MOV     RKASOF(R2),T.ASOF
10294  050066  016237  000020  003012          MOV     RKDCYL(R2),T.DC
10295  050074  016237  000026  003022          MOV     RKMR1(R2),T.MR1
10296  050102  016237  000034  003024          MOV     RKMR2(R2),T.MR2
10297  050110  016237  000036  003026          MOV     RKMR3(R2),T.MR3
10298  050116  016237  000030  003030          MOV     RKECPS(R2),T.POS
10299  050124  016237  000032  003032          MOV     RKECPT(R2),T.PAT
10300  050132  000207                          RTS     PC              ;RETURN
```

```
10301                                   .SBTTL   *STORE CONTROLLER STATUS
10302
10303                                   ;;*****************************************************************
10304                                   ;*
10305                                   ;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
10306                                   ;*      THE FOLLOWING REGISTERS WILL BE STORED:
10307                                   ;*
10308                                   ;*              COMMAND AND STATUS REGISTER 2
10309                                   ;*              WORD COUNT REGISTER
10310                                   ;*              BUS ADDRESS REGISTER
10311                                   ;*              DESIRED TRACK AND SECTOR
10312                                   ;*              STATUS REGISTER
10313                                   ;*              ERROR REGISTER
10314                                   ;*              ATTENTION SUMMARY/OFFSET REGISTER
10315                                   ;*              CYLINDER ADDRESS REGISTER
10316                                   ;*
10317                                   ;*CALL  JSR     PC,I.CSTS
10318                                   ;*      RETURN
10319                                   ;*
10320                                   ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
10321                                   ;*
10322                                   ;*
10323                                   ;*              REGISTER                CONTENTS
10324                                   ;*              --------                --------
10325                                   ;*
10326                                   ;*              R2                      RK06 BASE ADDRESS
10327                                   ;*              R5                      ADDRESS OF PARAMETER BLOCK
10328                                   ;*
10329                                   ;;*****************************************************************
10330
10331  050134  042765  177741  000016  I.CSTS: BIC     #177741,P.CS1(R5)       ;CLEAR ALL BITS EXCEPT FUNCTION
10332                                                                          ;OF LAST COMMAND ISSUED
10333  050142  042737  000036  003000          BIC     #36,T.CS1               ;CLEAR FUNCTION OF CS1 STATUS
10334  050150  053765  003000  000016          BIS     T.CS1,P.CS1(R5)         ;GENERATE CS1 STATUS INFORMATION
10335  050156  016265  000010  000020  I.CST1: MOV     RKCS2(R2),P.CS2(R5)     ;STORE COMMAND AND STATUS REGISTER 2
10336  050164  016265  000002  000022          MOV     RKWC(R2),P.WCR(R5)      ;STORE WORD COUNT REGISTER
10337  050172  016265  000004  000024          MOV     RKBA(R2),P.BAR(R5)      ;STORE BUS ADDRESS REGISTER
10338  050200  016265  000006  000026          MOV     RKDA(R2),P.DTS(R5)      ;STORE DESIRED TRACK AND SECTOR
10339  050206  016265  000012  000036          MOV     RKDS(R2),P.DS(R5)       ;STORE DRIVE STATUS REGISTER
10340  050214  016265  000014  000034          MOV     RKER(R2),P.ER(R5)       ;STORE ERROR REGISTER
10341  050222  016265  000016  000032          MOV     RKASOF(R2),P.ASOF(R5)   ;STORE ATTENTION SUMMARY AND
10342                                                                          ;  OFFSET
10343  050230  016265  000020  000030          MOV     RKDCYL(R2),P.DCYL(R5)   ;STORE CYLINDER ADDRESS
10344  050236  000207                          RTS     PC                      ;RETURN
10345
```

```
10346                                   .SBTTL   *GATHER DRIVE STATUS
10347
10348                                   ;;*****************************************************************
10349                                   ;*
10350                                   ;*      THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
10351                                   ;*      BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
10352                                   ;*      HAS PREVIOUSLY BEEN SEIZED.  IT RUNS AT PRIORITY 7.
10353                                   ;*
10354                                   ;*CALL  JSR     R0,I.STAT
10355                                   ;*      <ADDRESS OF ERROR RETURN>
10356                                   ;*      RETURN
10357                                   ;*
10358                                   ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
10359                                   ;*
10360                                   ;*
10361                                   ;*              REGISTER                CONTENTS
10362                                   ;*              --------                --------
10363                                   ;*
10364                                   ;*              R2                      RK06 BASE ADDRESS
10365                                   ;*              R5                      ADDRESS OF PARAMETER BLOCK
10366                                   ;*
10367                                   ;*      ROUTINES USED:
10368                                   ;*              I.ISSU
10369                                   ;*
10370                                   ;;*****************************************************************
10371
10372  050240  012762  000001  000026  I.STAT: MOV     #1,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
10373                                                                          ;  FOR STATUS BYTE 01
10374  050246  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10375  050254  004037  047564                  JSR     R0,I.ISSU               ;GET STATUS BYTES 01
10376  050260  050450                          3$                              ;ERROR RETURN
10377  050262  013765  003024  000044          MOV     T.MR2,P.A01(R5)         ;STORE STATUS BYTE 01 MESS A
10378  050270  013765  003026  000046          MOV     T.MR3,P.B01(R5)         ;STORE STATUS BYTE 01 MESS B
10379  050276  012762  000002  000026          MOV     #2,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER 1
10380                                                                          ;  FOR STATUS BYTE 10
10381  050304  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10382  050312  004037  047564                  JSR     R0,I.ISSU               ;GET STATUS BYTES 10
10383  050316  050450                          3$                              ;ERROR RETURN
10384  050320  013765  003024  000050          MOV     T.MR2,P.A10(R5)         ;STORE STATUS BYTE 10 MESS A
10385  050326  013765  003026  000052          MOV     T.MR3,P.B10(R5)         ;STORE STATUS BYTE 10 MESS B
10386  050334  012762  000003  000026          MOV     #3,RKMR1(R2)            ;LOAD MAINTENANCE REGISTER
10387                                                                          ;  FOR STATUS BYTE 11
10388  050342  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10389  050350  004037  047564                  JSR     R0,I.ISSU               ;GET STATUS BYTES 11
10390  050354  050450                          3$                              ;ERROR RETURN
10391  050356  013765  003024  000054          MOV     T.MR2,P.A11(R5)         ;STORE STATUS BYTE 11 MESS A
10392  050364  013765  003026  000056          MOV     T.MR3,P.B11(R5)         ;STORE STATUS BYTE 11 MESS B
10393  050372  005062  000026                  CLR     RKMR1(R2)               ;LOAD MAINTENANCE REGISTER 1
10394                                                                          ;  FOR STATUS BYTE 00
10395  050376  112737  000001  003000          MOVB    #DR.SEL,T.CS1           ;LOAD COMMAND
10396  050404  004037  047564                  JSR     R0,I.ISSU               ;GET STATUS BYTES 00
10397  050410  050450                          3$                              ;ERROR RETURN
10398  050412  013765  003024  000040          MOV     T.MR2,P.A00(R5)         ;STORE STATUS BYTE 00 MESS A
10399  050420  013765  003026  000042          MOV     T.MR3,P.B00(R5)         ;STORE STATUS BYTE 00 MESS B
10400  050426  032737  001000  003026          BIT     #S.PAR,T.MR3            ;CHECK IF BAD PARITY DETECTED BY DRIVE
10401  050434  001407                          BEQ     5$                      ;NO, RETURN NORMALLY
```

```
10402  050436  052737  002000  003052          BIS     #E.DPAR,E.CON*          ;INDICATE BAD PARITY DETECTED BY DRIVE
10403  050444  004737  050512                  JSR     PC,R.CONT               ;REPORT ERROR
10404  050450  011000                  3$:     MOV     (R0),R0                 ;LOAD R0 FOR ERROR RETURN
10405  050452  000200                          RTS     R0                      ;RETURN
10406
10407  050454  052765  001000  000014  5$:     BIS     #PBSVAL,P.PRST(R5)      ;SET PARAMETER BLOCK STATUS VALID
10408  050462  005720                          TST     (R0)+                   ;ADJUST R0 FOR NORMAL RETURN
10409  050464  000200                          RTS     R0                      ;RETURN
10410
```

```
                                .SBTTL   *COMMON DRIVER RETURNS

050466  105037  003101        R.ABNL: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
050472  004777  132350                JSR     PC,@A.ABNL      ;INDICATE ABNORMAL RETURN
050476  000207                        RTS     PC              ;RETURN

050500  105037  003101        R.NORM: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
050504  004777  132334                JSR     PC,@A.NORM      ;INDICATE NORMAL RETURN
050510  000207                        RTS     PC              ;RETURN

050512  105037  003101        R.CONT: CLRB    INTMSK          ;INHIBIT FUTURE DRIVE INTERRUPT REPORTING
050516  105037  003100                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
050522  005037  003114                CLR     W.DRV           ;CLEAR TIMING COUNT FOR THIS DRIVE
050526  004777  132316                JSR     PC,@A.CONT      ;INDICATE CONTROLLER ERROR RETURN
050532  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
                        ;*
                        ;;*****************************************************************

050534  010546          C.INIT: MOV     R5,-(SP)        ;STORE R5 ON STACK
050536  010446                  MOV     R4,-(SP)        ;STORE R4 ON STACK
050540  010346                  MOV     R3,-(SP)        ;STORE R3 ON STACK
050542  010246                  MOV     R2,-(SP)        ;STORE R2 ON STACK
050544  010146                  MOV     R1,-(SP)        ;STORE R1 ON STACK
050546  010046                  MOV     R0,-(SP)        ;STORE R0 ON STACK
050550  013746  177776          MOV     PS,-(SP)        ;STORE PSW ON STACK
050554  013737  003042  177776  MOV     RKPRI,PS        ;LOCK OUT RK06 INTERRUPTS
050562  017605  000016          MOV     @16(SP),R5      ;STORE PARAMETER BLOCK ADDRESS
050566  062766  000002  000016  ADD     #2,16(SP)       ;ADJUST RETURN
050574  016504  000000          MOV     P.DRVN(R5),R4   ;STORE DRIVE NUMBER
050600  042704  177770          BIC     #^C<DRVMSK>,R4  ;MASK OUT JUNK
050604  010537  00311?          MOV     R5,PBLKT        ;LOAD PARAMETER BLOCK TABLE
050610  116437  003102  003101  MOVB    I.DRV(R4),INTMSK ;LOAD INTERRUPT MASK
050616  116437  003102  003100  MOVB    I.DRV(R4),W.TIME ;SET WATCH-DOG TIMER FLAG
050624  013737  003062  003114  MOV     W.SEC,W.DRV     ;LOAD WATCH-DOG TIME

050632  013702  003036          MOV     RKBAS,R2        ;LOAD R2 WITH RK06 ADDRESS BASE

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

```
10482  050636  042765  075176  000014          BIC     #^C<DRVUSE!W.WCK!NOCHK!DRPDRV!DTBAII>,P.PRST(R5)
10483
10484  050644  010500                          MOV     R5,R0           ;STORE PARAMETER BLOCK ADDRESS
10485  050646  062700  000016                  ADD     #P.CS1,R0       ;CALCULATE FIRST LOCATION TO BE CLEARED
10486  050652  010501                          MOV     R5,R1           ;STORE PARAMETER BLOCK ADDRESS
10487  050654  062701  000062                  ADD     #P.EPAT,R1      ;CALCULATE LAST LOCATION TO BE CLEARED
10488
10489  050660  005020                  1$:     CLR     (R0)+           ;CLEAR RETURN PARAMETER
10490  050662  020001                          CMP     R0,R1           ;CHECK IF FINISHED
10491  050664  101775                          BLOS    1$              ;NO, CLEAR NEXT RETURN PARAMETER
10492  050666  105037  003074                  CLRB    I.ISRL          ;CLEAR RELEASE OR INTERRUPT ISSUED
10493  050672  010465  000020                  MOV     R4,P.CS2(R5)    ;STORE DRIVE NUMBER
10494  050676  005062  000026                  CLR     RKMR1(R2)       ;CLEAR RK06 MAINTENANCE REGISTER 1
10495  050702  132765  000040  000001          BITB    #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
10496  050710  001402                          BEQ     3$              ;NO, PROCESS
10497  050712  000137  051426                  JMP     C.SPEC          ;JUMP TO SPECIAL COMMAND PROCESSOR
10498
10499  050716  122765  000107  000001  3$:     CMPB    #UNLOAD,P.CMND(R5) ;CHECK IF POSITIONING COMMAND
10500                                                                  ;  START SPINDLE
10501                                                                  ;  RECALIBRATE
10502                                                                  ;  OFFSET
10503                                                                  ;  SEEK
10504                                                                  ;  UNLOAD
10505
10506  050724  101174                          BHI     25$             ;NO, DRIVE COMMAND
10507                                                                  ;  SELECT DRIVE
10508                                                                  ;  PACK ACKNOWLEDGE
10509                                                                  ;  CLEAR
10510
10511  050726  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF DATA TRANSFER
10512  050734  103540                          BLO     20$             ;YES, DATA TRANSFER COMMAND
10513                                                                  ;  READ DATA
10514                                                                  ;  WRITE DATA
10515                                                                  ;  READ HEADER
10516                                                                  ;  WRITE HEADER
10517                                                                  ;  WRITE CHECK
10518  050736  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
10519  050744  052765  000002  000014          BIS     #DRVPOS,P.PRST(R5) ;SET DRIVE POSITIONING
10520  050752  005037  003054                  CLR     O.WAIT          ;CLEAR WAIT FOR COMMAND
10521  050756  122765  000117  000001          CMPB    #SEEK,P.CMND(R5) ;CHECK IF SEEK
10522  050764  001007                          BNE     5$              ;NO, CHECK FOR OFFSET
10523  050766  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
10524  050774  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK
10525  051002  000431                          BR      8$              ;GO ISSUE COMMAND
10526
10527  051004  122765  000115  000001  5$:     CMPB    #OFFSET,P.CMND(R5) ;CHECK IF OFFSET
10528  051012  001007                          BNE     6$              ;NO, CHECK FOR UNLOAD
10529  051014  116565  000006  000032          MOVB    P.OFST(R5),P.ASOF(R5) ;STORE OFFSET
10530  051022  016562  000032  000016          MOV     P.ASOF(R5),RKASOF(R2) ;LOAD OFFSET REGISTER
10531  051030  000416                          BR      8$              ;GO ISSUE COMMAND
10532
10533  051032  122765  000111  000001  6$:     CMPB    #SRTSPL,P.CMND(R5) ;CHECK IF START SPINDLE
10534  051040  001003                          BNE     7$              ;NO, CHECK IF RECAL
10535  051042  013737  003066  003114          MOV     W.MIN,W.DRV     ;LOAD WATCH DOG TIME FOR 1 MINUTE
10536  051050  122765  000113  000001  7$:     CMPB    #RECAL,P.CMND(R5) ;CHECK IF RECAL
10537  051056  001003                          BNE     8$              ;NO, CONTINUE
```

```
10538  051060  013737  003064  003114          MOV     W.BSEC,W.DRV     ;LOAD RECAL TIME FOR 8 SECONDS
10539  051066  116565  000007  000017  8$:     MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10540  051074  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10541                                                                  ;  AND DRIVE TYPE
10542  051102  116565  000001  000016          MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10543  051110  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10544  051116  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10545  051124  001533                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
10546  051126  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
10547  051134  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10548  051142  004737  045104          10$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
10549  051146  016237  000000  003000          MOV     RKCS1(R2),T.CS1  ;STORE COMMAND AND STATUS REGISTER 1
10550  051154  032737  000200  003000          BIT     #RDY,T.CS1       ;WAIT FOR READY
10551  051162  001767                          BEQ     10$
10552  051164  032737  100000  003000          BIT     #CERR,T.CS1      ;CHECK FOR ERROR
10553  051172  001011                          BNE     15$              ;YES, GIVE NORMAL RETURN
10554  051174  004737  045104          11$:    JSR     PC,W.WTCH        ;CALL WATCH DOG TIMER
10555  051200  016237  000016  003014          MOV     RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
10556  051206  133737  003101  003015          BITB    INTMSK,T.ASOF+1 ;CHECK IF INTERRUPT HAS OCCURRED
10557  051214  001767                          BEQ     11$              ;WAIT FOR DRIVE INTERRUPT
10558  051216  105037  003100          15$:    CLRB    W.TIME           ;RESET TIMING ON THIS DRIVE
10559  051222  005037  003114                  CLR     W.DRV            ;CLEAR DRIVE TIMING COUNT
10560  051226  004737  050500                  JSR     PC,R.NORM        ;INDICATE COMMAND IS FINISHED
10561  051232  000137  052406                  JMP     C.RTRN           ;RESTORE REGISTERS
10562
10563  051236  016562  000010  000004  20$:    MOV     P.BALO(R5),RKBA(R2) ;LOAD BUS ADDRESS REGISTER
10564  051244  016562  000012  000002          MOV     P.WC(R5),RKWC(R2) ;LOAD WORD COUNT REGISTER
10565  051252  016562  000002  000020          MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
10566  051260  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD SECTOR AND TRACK NUMBER
10567  051266  122765  000131  000001          CMPB    #WRTCHK,P.CMND(R5)  ;CHECK IF WRITE CHECK COMMAND
10568  051274  001010                          BNE     25$              ;NO, GO ISSUE THE COMMAND
10569  051276  032765  000200  000014          BIT     #W.WCK,P.PRST(R5)  ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
10570  051304  001404                          BEQ     25$              ;NO, GO ISSUE THE COMMAND
10571  051306  012765  000123  000016          MOV     #WRDATA,P.CS1(R5)  ;ISSUE WRITE COMMAND
10572  051314  000406                          BR      26$              ;GO ISSUE COMMAND
10573
10574  051316  116565  000001  000016  25$:    MOVB    P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
10575  051324  042765  000200  000014          BIC     #W.WCK,P.PRST(R5) ;RESET WRITE FOR WRITE CHECK
10576  051332  116565  000007  000017  26$:    MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10577  051340  142765  177750  000017          BICB    #^C<B.CFMT!B.CDT!B.BA16!B.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
10578                                                                  ;  FORMAT, DRIVE TYPE, AND BUS ADDRESS
10579                                                                  ;  BITS 16-17
10580  051346  010537  003054                  MOV     R5,O.WAIT        ;LOAD WAITING FOR COMMAND
10581  051352  032765  100000  000014          BIT     #DTBAII,P.PRST(R5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
10582  051360  001403                          BEQ     27$              ;NO, LOAD CS2
10583  051362  052765  000020  000020          BIS     #BAI,P.CS2(R5)   ;SET INHIBIT BUS ADDRESS INCREMENT
10584  051370  016562  000020  000010  27$:    MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2
10585  051376  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IN NO CHECK MODE
10586  051404  001403                          BEQ     30$              ;NO, SKIP CLEAR OF INTERRUPT ENABLE
10587  051406  042765  000100  000016          BIC     #IE,P.CS1(R5)    ;CLEAR INTERRUPT ENABLE
10588  051414  016562  000016  000000  30$:    MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10589  051422  000137  052406                  JMP     C.RTRN           ;RESTORE REGISTERS
10590
10591                                  .SBTTL   *SPECIAL COMMAND PROCESSING
10592
10593  051426  122765  000141  000001  C.SPEC: CMPB    #RDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
```

```
10594  051434  001132                        BNE     10$             ;NO, PROCESS OTHER COMMANDS
10595  051436  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR COMMAND
10596  051444  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
10597  051452  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10598                                                                ;  AND DRIVE TYPE
10599  051460  112765  000001  000016        MOVB    #DR.SEL,P.CS1(R5) ;STORE COMMAND
10600  051466  016562  000016  000000        MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
10601  051474  004737  045104          2$:   JSR     PC,W.WTCH       ;CALL WATCH-DOG TIMER
10602  051500  016265  000000  000016        MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
10603  051506  032765  000200  000016        BIT     #RDY,P.CS1(R5)  ;WAIT FOR READY
10604  051514  001767                        BEQ     2$
10605  051516  004737  050156                JSR     PC,I.CST1       ;STORE CONTROLLER REGISTERS
10606  051522  016265  000034  000040        MOV     RKMR2(R2),P.A00(R5) ;STORE STATUS BYTE 00 MESSAGE A
10607  051530  016265  000036  000042        MOV     RKMR3(R2),P.B00(R5) ;STORE STATUS BYTE 00 MESSAGE B
10608  051536  032765  100000  000016        BIT     #CERR,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
10609  051544  001436                        BEQ     6$              ;NO, GATHER DRIVE STATUS
10610  051546  105037  003100                CLRB    W.TIME          ;RESET WATCH DOG TIMING ON THIS DRIVE
10611  051552  005037  003114                CLR     W.DRV           ;CLEAR WATCH DOG COUNT
10612  051556  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10613  051564  001043                        BNE     8$              ;YES, INDICATE NORMAL RETURN
10614  051566  032765  001000  000020        BIT     #MDS,P.CS2(R5)  ;CHECK IF MULTIPLE DRIVE SELECT
10615  051574  001043                        BNE     9$              ;YES, INDICATE CONTROLLER ERROR
10616  051576  004037  047502                JSR     R0,I.CCLR       ;CLEAR ERROR
10617  051602  052406                        C.RTRN                  ;ERROR RETURN
10618  051604  032765  010400  000020        BIT     #NED!UFE,P.CS2(R5) ;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
10619  051612  001007                        BNE     5$              ;REPORT ERROR
10620  051614  032765  000001  000036        BIT     #DRA,P.DS(R5)   ;CHECK IF DRIVE AVAILIABLE
10621  051622  001003                        BNE     5$              ;YES, REPORT ERROR
10622  051624  052765  010000  000014        BIS     #DRVSZD,P.PRST(R5) ;INDICATE DRIVE IS SEIZED BY OTHER PORT
10623  051632  004737  050466          5$:   JSR     PC,R.ABNL       ;INDICATE ABNORMAL RETURN
10624  051636  000137  052406                JMP     C.RTRN          ;RESTORE REGISTERS
10625
10626  051642  004037  050240          6$:   JSR     R0,I.STAT       ;GATHER DRIVE STATUS
10627  051646  052406                        C.RTRN                  ;ERROR RETURN
10628  051650  105037  003100                CLRB    W.TIME          ;STOP WATCH-DOG TIMING ON DRIVE
10629  051654  005037  003114                CLR     W.DRV           ;RESET WATCH-DOG TIME
10630  051660  032765  000400  000014        BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10631  051666  001402                        BEQ     8$              ;NO, REPORT ERROR
10632  051670  005062  000000                CLR     RKCS1(R2)       ;CLEAR INTERRUPT ENABLE
10633  051674  004737  050500          8$:   JSR     PC,R.NORM       ;REPORT COMMAND COMPLETE
10634  051700  000137  052406                JMP     C.RTRN          ;RESTORE REGISTERS
10635
10636  051704  052737  100000  003052  9$:   BIS     #E.MDS,E.CONT   ;SET MULTIPLE DRIVE SELECT
10637  051712  004737  050512                JSR     PC,R.CONT       ;INDICATE CONTROLLER ERROR
10638  051716  000137  052406                JMP     C.RTRN
10639
10640  051722  122765  000140  000001  10$:   CMPB    #RELEAS,P.CMND(R5) ;CHECK IF RELEASE COMMAND
10641  051730  001040                        BNE     13$             ;NO, CHECK IF READ ALL HEADERS
10642  051732  010537  003054                MOV     R5,O.WAIT       ;STORE PARAMETER BLOCK ADDRESS IN
10643                                                                ;  WAIT FOR COMMAND
10644  051736  052765  000010  000020        BIS     #RLS,P.CS2(R5)  ;SET RELEASE BIT
10645  051744  016562  000020  000010        MOV     P.CS2(R5),RKCS2(R2) ;LOAD CS2 FOR DESELECT
10646  051752  112737  000001  003074        MOVB    #1,I.ISRL       ;SET FLAG FOR RELEASE COMMAND
10647  051760  116565  000007  000017        MOVB    P.CS1H(R5),P.CS1+1(R5);STORE BITS 8-15 OF CS1
10648  051766  042765  165777  000016        BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FORMAT
10649                                                                ;  AND DRIVE TYPE
```

```
 10650  051774  112765  000101  000016          MOVB    #SELDRV,P.CS1(R5) ;STORE COMMAND
 10651  052002  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10652  052010  001403                          BEQ     11$               ;NO, DO NOT RESET INTERRUPT ENABLE
 10653  052012  042765  000100  000016          BIC     #IE,P.CS1(R5)     ;RESET INTERRUPT ENABLE
 10654  052020  016562  000016  000000  11$:     MOV     P.CS1(R5),RKCS1(R2) ;ISSUE COMMAND
 10655  052026  000137  052406                  JMP     C.RTRN            ;RESTORE REGISTERS
 10656
 10657  052032  122765  000164  000001  13$:     CMPB    #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
 10658  052040  001053                          BNE     30$               ;NO, CHECK IF CONTROLLER CLEAR
 10659  052042  010537  003054                  MOV     R5,D.WAIT         ;SET WAITING FOR COMMAND COMMPLETION
 10660  052046  016537  000010  003070          MOV     P.BALO(R5),HDR.AD ;LOAD HEADER ADDRESS
 10661  052054  132765  000020  000007          BITB    #B.CFMT,P.CS1H(R5) ;CHECK IF 22 SECTOR FORMANT
 10662  052062  001404                          BEQ     14$               ;YES, LOAD 22 IN HEADER COUNT
 10663  052064  012737  000024  003072          MOV     #20.,HDR.CT       ;LOAD 20 IN SECTOR COUNT
 10664  052072  000403                          BR      22$               ;GO ISSUE READ HEADER COMMAND
 10665
 10666  052074  012737  000026  003072  14$:     MOV     #22.,HDR.CT       ;LOAD 22 IN SECTOR COUNT
 10667  052102  016562  000002  000020  22$:     MOV     P.CYLN(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
 10668  052110  016562  000004  000006          MOV     P.SECT(R5),RKDA(R2) ;LOAD TRACK NUMBER
 10669  052116  016562  000020  000010          MOV     P.CS2(R5),RKCS2(R2) ;LOAD DRIVE NUMBER
 10670  052124  116565  000007  000017          MOVB    P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 8-15 OF CS1
 10671  052132  042765  165777  000016          BIC     #^C<CFMT!CDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT DRIVE TYPE
 10672                                                                  ;  AND FORMAT
 10673  052140  112765  000125  000016          MOVB    #RDHEAD,P.CS1(R5) ;STORE READ HEADER COMMAND
 10674  052146  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10675  052154  001027                          BNE     34$               ;YES, INDICATE ILLEGAL DRIVER COMMAND
 10676  052156  016562  000016  000000          MOV     P.CS1(R5),RKCS1(R2) ;ISSUE READ HEADER
 10677  052164  000137  052406                  JMP     C.RTRN            ;RESTORE REGISTERS
 10678
 10679  052170  122765  000176  000001  30$:     CMPB    #CONCLR,P.CMND(R5) ;CHECK IF CONTROLLER   EAR
 10680  052176  001012                          BNE     32$               ;NO, CHECK IF SUBSYSTEM   EAR
 10681  052200  004037  047502                  JSR     R0,I.CCLR         ;CLEAR CONTROLLER
 10682  052204  052406                          C.RTRN                    ;ERROR RETURN
 10683  052206  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
 10684  052214  001472                          BEQ     40$               ;NO, INDICATE NORMAL RETURN
 10685  052216  005062  000000                  CLR     RKCS1(R2)         ;RESET INTERRUPT ENABLE
 10686  052222  000467                          BR      40$               ;INDICATE NORMAL RETURN
 10687
 10688  052224  122765  000177  000001  32$:     CMPB    #SUBCLR,P.CMND(R5) ;CHECK IF SUBSYSTEM CLEAR
 10689  052232  001406                          BEQ     36$               ;YES, CLEAR SUBSYSTEM
 10690  052234  052737  000100  003052  34$:     BIS     #E.ILLD,E.CONT    ;SET ILLEGAL DRIVER COMMAND
 10691  052242  004737  050512                  JSR     PC,R.CONT         ;REPORT ERROR
 10692  052246  000457                          BR      C.RTRN            ;RESTORE REGISTERS
 10693
 10694  052250  012762  000040  000010  36$:     MOV     #SCLR,RKCS2(R2) ;ISSUE SUBSYSTEM CLEAR
 10695  052256  016265  000000  000016          MOV     RKCS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REGISTER 1
 10696  052264  032765  100000  000016          BIT     #CERR,P.CS1(R5) ;CLEAR IF CONTROLLER ERROR RESET
 10697  052272  001406                          BEQ     37$               ;NO, FINISH COMMAND
 10698  052274  052737  000001  003052          BIS     #BIT0,E.CONT      ;SET CLEAR SUBSYSTEM DID NOT CLEAR
 10699                                                                  ;  CONTROLLER ERROR
 10700  052302  004737  050512                  JSR     PC,R.CONT         ;REPORT ERROR
 10701  052306  000437                          BR      C.RTRN            ;RESTORE REGISTERS
 10702
 10703  052310  013746  003060          37$:     MOV     W.MILI,-(SP)      ;LOAD 16 MILI-SECOND COUNT FOR ATTENTION
 10704                                                                  ;  TO DISAPPEAR
 10705  052314  016265  000000  000016  38$:     MOV     RKCS1(R2),P.CS1(R5) ;STORE CS1
```

```
10706  052322  032765  040000  000016          BIT     #DI,P.CS1(R5)   ;CHECK IF ATTENTIONS CLEARED
10707  052330  001411                          BEQ     39$             ;YES, FINISH COMMAND
10708  052332  005316                          DEC     (SP)            ;DECREMENT 16 MILISECOND COUNT
10709  052334  001367                          BNE     38$             ;CHECK DRIVE INTERRUPT AGAIN
10710  052336  005726                          TST     (SP)+           ;ADJUST STACK
10711  052340  052737  000040  003052          BIS     #E.SCLR,E.CONT  ;SET SUBSYSTEM CLEAR DID NOT CLEAR
10712                                                                  ;  DRIVE ATTENTIONS
10713  052346  004737  050512                  JSR     PC,R.CONT       ;REPORT ERROR
10714  052352  000415                          BR      C.RTRN          ;RESTORE REGISTER
10715
10716  052354  005726                  39$:    TST     (SP)+           ;ADJUST STACK
10717  052356  032765  000400  000014          BIT     #NOCHK,P.PRST(R5) ;CHECK IF NO CHECK MODE
10718  052364  001010                          BNE     C.RTRN          ;YES, RESTORE REGISTERS
10719  052366  112737  177777  003074          MOVB    #-1,I.ISRL      ;SET INTERRUPT ENABLE SET
10720  052374  012762  000100  000000          MOV     #IE,RKCS1(R2)   ;SET INTERRUPT ENABLE
10721  052402  004737  050500          40$:    JSR     PC,R.NORM       ;INDICATE NORMAL TERMINATION
10722
10723  052406  012637  177776          C.RTRN: MOV     (SP)+,PS        ;RESTORE PSW
10724  052412  012600                          MOV     (SP)+,R0        ;RESTORE R0
10725  052414  012601                          MOV     (SP)+,R1        ;RESTORE R1
10726  052416  012602                          MOV     (SP)+,R2        ;RESTORE R2
10727  052420  012603                          MOV     (SP)+,R3        ;RESTORE R3
10728  052422  012604                          MOV     (SP)+,R4        ;RESTORE R4
10729  052424  012605                          MOV     (SP)+,R5        ;RESTORE R5
10730  052426  000207                          RTS     PC              ;RETURN
10731                                  .SBTTL  OCTAL TO BINARY CONVERSION ROUTINE
10732
10733                                  ;;**********************************************************************
10734                                  ;*
10735                                  ;*      THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
10736                                  ;*      WITH A NULL <000> OR COMMA.  IF THE CHARACTERS ARE LEGAL
10737                                  ;*      IT  WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
10738                                  ;*      ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HIOCT.
10739                                  ;*
10740                                  ;*CALL
10741                                  ;*      MOV     <ADDRESS OF ASCII STRING>,-(SP)
10742                                  ;*      JSR     PC,OCTBIN
10743                                  ;*      <ADDRESS OF ERROR RETURN>
10744                                  ;*      RETURN
10745                                  ;*
10746                                  ;;**********************************************************************
10747
10748  052430  010046                  OCTBIN: MOV     R0,-(SP)        ;SAVE R0
10749  052432  010146                          MOV     R1,-(SP)        ;SAVE R1
10750  052434  010246                          MOV     R2,-(SP)        ;SAVE R2
10751  052436  016600  000010                  MOV     10(SP),R0       ;GET ADDRESS OF ASCII STRING
10752  052442  005001                          CLR     R1              ;CLEAR DATA WORDS
10753  052444  005002                          CLR     R2
10754  052446  112046                  2$:     MOVB    (R0)+,-(SP)     ;PICK THIS CHARACTER
10755  052450  001423                          BEQ     3$              ;IF ZERO GET OUT
10756  052452  121627  000054                  CMPB    (SP),#',        ;CHECK IF COMMA
10757  052456  001420                          BEQ     3$              ;IF COMMA GET OUT
10758  052460  122716  000060                  CMPB    #'0,(SP)        ;MAKE SURE THIS CHARACTER IS
10759  052464  003030                          BGT     4$              ;  AN OCTAL DIGIT
10760  052466  122716  000067                  CMPB    #'7,(SP)
10761  052472  002425                          BLT     4$
```

```
10762  052474  006301                  ASL     R1              ; *2
10763  052476  006102                  ROL     R2
10764  052500  006301                  ASL     R1              ; *4
10765  052502  006102                  ROL     R2
10766  052504  006301                  ASL     R1              ; *8
10767  052506  006102                  ROL     R2
10768  052510  042716  177770          BIC     #^C7,(SP)       ;STRIP THE ASCII JUNK
10769  052514  062601                  ADD     (SP)+,R1        ;ADD THIS DIGIT
10770  052516  000753                  BR      2$              ;LOOP
10771  052520  005726          3$:     TST     (SP)+           ;CLEAN PARTIAL FROM STACK
10772  052522  010166  000010          MOV     R1,10(SP)       ;SAVE RESULT
10773  052526  010237  052562          MOV     R2,$HIOCT
10774  052532  012602                  MOV     (SP)+,R2        ;RESTORE R2
10775  052534  012601                  MOV     (SP)+,R1        ;RESTORE R1
10776  052536  012600                  MOV     (SP)+,R0        ;RESTORE R0
10777  052540  062716  000002          ADD     #2,(SP)         ;ADJUST RETURN
10778  052544  000207                  RTS     PC              ;RETURN
10779
10780  052546  005726          4$:     TST     (SP)+           ;CLEAN UP PARTIAL FROM STACK
10781  052550  012602                  MOV     (SP)+,R2        ;RESTORE R2
10782  052552  012601                  MOV     (SP)+,R1        ;RESTORE R1
10783  052554  012600                  MOV     (SP)+,R0        ;RESTORE R0
10784  052556  013616                  MOV     @(SP)+,(SP)     ;PUT ADDRESS OF ERROR ROUTINE ON STACK
10785  052560  000207                  RTS     PC              ;GO PROCESS ERROR
10786  052562  000000          $HIOCT: .WORD   0               ;HIGH ORDER BITS GO HERE
10787                           .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
10788
10789                           ;;**************************************************************************
10790                           ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
10791                           ;*WITH A RANGE OF 0 TO 2(+33)-1.
10792                           ;*CALL:
10793                           ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
10794                           ;*      RETURN                  ;;RETURN HERE THE RANDOM
10795                           ;*                              ;;NUMBER WILL BE IN
10796                           ;*                              ;;$HINUM,$LONUM
10797
10798  052564                   $RAND:
10799  052564  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
10800  052566  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
10801  052570  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
10802  052572  013700  052664          MOV     $LONUM,R0       ;;SET R0 WITH LOW
10803  052576  013701  052662          MOV     $HINUM,R1       ;;SET R1 WITH HIGH
10804  052602  012702  177771          MOV     #-7,R2          ;;SET SHIFT COUNT
10805  052606  006300          1$:     ASL     R0              ;;SHIFT R0 LEFT AND
10806  052610  006101                  ROL     R1              ;;ROTATE CARRY INTO R1 AND
10807  052612  005202                  INC     R2              ;;CHECK FOR DONE
10808  052614  001374                  BNE     1$              ;;CONTINUE SHIFT LOOP
10809  052616  063700  052664          ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
10810  052622  005501                  ADC     R1              ;;PROPOGATE CARRY
10811  052624  063701  052662          ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
10812  052630  062700  001057          ADD     #1057,R0        ;;ADD LOW CONSTANT
10813  052634  005501                  ADC     R1              ;;PROPOGATE CARRY
10814  052636  062701  047401          ADD     #47401,R1       ;;ADD HIGH CONSTANT
10815  052642  010037  052664          MOV     R0,$LONUM       ;;SAVE R0
10816  052646  010137  052662          MOV     R1,$HINUM       ;;SAVE R1
10817  052652  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
10818  052654  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
10819  052656  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
10820  052660  000207                  RTS     PC              ;;RETURN
10821  052662  176543          $HINUM: .WORD   176543
10822  052664  123456          $LONUM: .WORD   123456
10823                           .SBTTL  TYPE ROUTINE
10824
10825                           ;;*****************************************************************
10826                           ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10827                           ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10828                           ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10829                           ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10830                           ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10831                           ;*
10832                           ;*CALL:
10833                           ;*1) USING A TRAP INSTRUCTION
10834                           ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
10835                           ;*OR
10836                           ;*      TYPE
10837                           ;*      MESADR
10838                           ;*
10839
10840  052666  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
10841  052672  100002                  BPL     1$              ;;BR IF YES
10842  052674  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
10843  052676  000430                  BR      3$              ;;LEAVE
10844  052700  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
10845  052702  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
10846  052706  122737  000001  001340          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
10847  052714  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
10848  052716  132737  000100  001341          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
10849  052724  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
10850  052726  010037  052736                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
10851  052732  004737  055364                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
10852  052736  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
10853  052740  132737  000040  001341  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
10854  052746  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
10855  052750  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
10856  052752  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
10857  052754  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
10858  052756  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
10859  052760  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
10860  052764  000002                          RTI                     ;;RETURN
10861  052766  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
10862  052772  001430                          BEQ     8$
10863  052774  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
10864  053000  001006                          BNE     5$
10865  053002  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
10866  053004  104401                          TYPE                    ;;TYPE A CR AND LF
10867  053006  001315                          $CRLF
10868  053010  105037  053144                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
10869  053014  000755                          BR      2$              ;;GET NEXT CHARACTER
10870  053016  004737  053100          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
10871  053022  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
10872  053026  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
10873  053030  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
```

```
10874                                                           ;;AND THE NULL CHAR.
10875  053034  105366  000001          7$:     DECB    1(SP)    ;;DOES A NULL NEED TO BE TYPED?
10876  053040  002770                          BLT     6$       ;;BR IF NO--GO POP THE NULL OFF OF STACK
10877  053042  004737  053100                  JSR     PC,$TYPEC ;;GO TYPE A NULL
10878  053046  105337  053144                  DECB    $CHARCNT ;;DO NOT COUNT AS A COUNT
10879  053052  000770                          BR      7$       ;;LOOP
10880
10881                                  ;HORIZONTAL TAB PROCESSOR
10882
10883  053054  112716  000040          8$:     MOVB    #' ,(SP) ;;REPLACE TAB WITH SPACE
10884  053060  004737  053100          9$:     JSR     PC,$TYPEC ;;TYPE A SPACE
10885  053064  132737  000007  053144          BITB    #7,$CHARCNT ;;BRANCH IF NOT AT
10886  053072  001372                          BNE     9$       ;;TAB STOP
10887  053074  005726                          TST     (SP)+    ;;POP SPACE OFF STACK
10888  053076  000724                          BR      2$       ;;GET NEXT CHARACTER
10889  053100  105777  126044          $TYPEC: TSTB    @$TPS    ;;WAIT UNTIL PRINTER IS READY
10890  053104  100375                          BPL     $TYPEC
10891  053106  116677  000002  126036          MOVB    2(SP),@$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
10892  053114  122766  000015  000002          CMPB    #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
10893  053122  001003                          BNE     1$       ;;BRANCH IF NO
10894  053124  105037  053144                  CLRB    $CHARCNT ;;YES--CLEAR CHARACTER COUNT
10895  053130  000406                          BR      $TYPEX   ;;EXIT
10896  053132  122766  000012  000002  1$:     CMPB    #LF,2(SP) ;;IS CHARACTER A LINE FEED?
10897  053140  001402                          BEQ     $TYPEX   ;;BRANCH IF YES
10898  053142  105227                          INCB    (PC)+    ;;COUNT THE CHARACTER
10899  053144  000000          $CHARCNT:.WORD  0                ;;CHARACTER COUNT STORAGE
10900  053146  000207          $TYPEX: RTS     PC
10901
10902
10903
10904
10905                                  ;*****************************************************************
10906                                  .SBTTL  DOUBLE-PRECISION MULTIPLY SUBROUTINE
10907                                  ;*      SUBROUTINE TO MULTIPLY TWO DOUBLE PRECISION INTEGERS
10908                                  ;*      USES ALL REGISTERS (R0-R5)
10909                                  ;*
10910                                  ;*      ENTER WITH   JSR   PC,M.DPIM
10911                                  ;*              MULTIPLIER IN R2-R3
10912                                  ;*              MULTIPLICAND IN R4-R5
10913                                  ;*              PRODUCT RETURNED IN R0-R1-R2-R3
10914                                  ;*
10915                                  ;*****************************************************************
10916  053150  005000          M.DPIM: CLR     R0               ;CLEAR HI ORDER WORDS
10917  053152  005001                  CLR     R1
10918  053154  012746  000041                  MOV     #41,-(SP) ;MOVE 33 (DEC) TO COUNTER
10919  053160  006000          M.DP01: ROR     R0
10920  053162  006001                  ROR     R1
10921  053164  006002                  ROR     R2               ;SHIFT TO ADD
10922  053166  006003                  ROR     R3
10923  053170  103003                  BCC     M.DP02           ;NO CARRY NO ADD
10924  053172  060501                  ADD     R5,R1
10925  053174  005500                  ADC     R0               ;ADD DOUBLE PRECISION TO OBTAIN NEW PARTIAL
10926  053176  060400                  ADD     R4,R0            ;  PRODUCT
10927  053200  005316          M.DP02: DEC     @SP              ;DECREMENT COUNTER
10928  053202  001366                  BNE     M.DP01
10929  053204  005726                  TST     (SP)+            ;REMOVE THE COUNTER
```

```
10930  053206  000207                    RTS     PC
10931
10932
10933                                ;;*****************************************************************
10934                                .SBTTL  DOUBLE-PRECISION DIVIDE SUBROUTINE
10935                                ;*              SUBROUTINE TO PERFORM DOUBLE-PRECISION INTEGER DIVISION
10936                                ;*              USES ALL REGISTERS (R0-R5)
10937                                ;*
10938                                ;*              ENTER WITH  JSR   PC,M.DPID
10939                                ;*                      DIVIDEND IN R0-R1-R2-R3
10940                                ;*                      DIVISOR IN R4-R5
10941                                ;*                      REMAINDER RETURNED IN R0-R1
10942                                ;*                      QUOTIENT RETURNED IN R2-R3
10943                                ;*
10944                                ;*****************************************************************
10945  053210  012746  000040       M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
10946  053214  010446                       MOV     R4,-(SP)        ;HI ORDER
10947  053216  010546                       MOV     R5,-(SP)        ;LO ORDER DIVISOR TO THE STACK
10948  053220  005466  000002               NEG     2(SP)           ;FORM NEGATIVE
10949  053224  005416                       NEG     @SP             ;   VERSION OF THE DIVISOR
10950  053226  005666  000002               SBC     2(SP)
10951  053232  061601                       ADD     @SP,R1
10952  053234  005500                       ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
10953  053236  066600  000002               ADD     2(SP),R0
10954  053242  103445                       BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
10955  053244  005046                       CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
10956  053246  006103       M.DP40: ROL     R3
10957  053250  006102               ROL     R2
10958  053252  006101               ROL     R1
10959  053254  006100               ROL     R0
10960  053256  005716               TST     @SP             ;TEST "CARRY INDICATOR"
10961  053260  001410               BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
10962  053262  005016               CLR     @SP             ;CLEAR UP FOR NEXT TIME
10963  053264  066601  000002       ADD     2(SP),R1
10964  053270  005500               ADC     R0              ;ADD -(DIVISOR)
10965  053272  005516               ADC     @SP     ; I     ;SET "CARRY"
10966  053274  066600  000004       ADD     4(SP),R0 ;<
10967  053300  000404               BR      M.DP42
10968  053302  060501       M.DP41: ADD     R5,R1
10969  053304  005500               ADC     R0              ;ADD +(DIVISOR)
10970  053306  005516               ADC     @SP     ; I     ;SET "CARRY"
10971  053310  060400               ADD     R4,R0   ;<
10972  053312  005516       M.DP42: ADC     @SP             ;SET "CARRY"
10973  053314  005716               TST     @SP             ;TEST THE UPDATE INDICATOR
10974  053316  001401               BEQ     .+4     ;>      ;IF ZERO FORGET IT
10975  053320  005203               INC     R3      ; I     ;NO CARRY POSSIBLE HERE
10976  053322  005366  000006       DEC     6(SP)   ;<      ;DECREMENT COUNTER
10977  053326  003347               BGT     M.DP40          ;BR IF MORE TO DO
10978  053330  006003               ROR     R3
10979  053332  103404               BCS     M.DP44
10980  053334  060501               ADD     R5,R1
10981  053336  005500               ADC     R0
10982  053340  060400               ADD     R4,R0
10983  053342  000241               CLC
10984  053344  006103       M.DP44: ROL     R3
10985  053346  062706  000010       ADD     #10,SP          ;ADJUST STACK BY 4 WORDS
```

```
10986  053352  000242                          CLV
10987  053354  000207                          RTS     PC
10988  053356  062706  000006          M.DP50: ADD     #6,SP
10989  053362  000262                          SEV
10990  053364  000207                          RTS     PC
10991
10992
10993
10994                                  .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
10995
10996                                  ;;**********************************************************************
10997                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
10998                                  ;*UNSIGNED OCTAL ASCIZ NUMBER.
10999                                  ;*CALL
11000                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
11001                                  ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
11002                                  ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
11003
11004
11005  053366  104407                  $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
11006  053370  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
11007  053374  012705  053505                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
11008  053400  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
11009  053404  012703  177770                  MOV     #^C7,R3         ;;MASK
11010  053410  012100                          MOV     (R1)+,R0        ;;LOWER WORD
11011  053412  012101                          MOV     (R1)+,R1        ;;HIGH WORD
11012  053414  005002                          CLR     R2              ;;TERMINATOR
11013  053416  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
11014  053420  010002                          MOV     R0,R2           ;;GET THIS DIGIT
11015  053422  005304                          DEC     R4              ;;COUNT THIS CHARACTER
11016  053424  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
11017  053426  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
11018  053430  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
11019  053432  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
11020  053436  104410                          RESREG                  ;;RESTORE ALL REGISTERS
11021  053440  000207                          RTS     PC              ;;RETURN TO USER
11022  053442  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
11023  053444  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
11024  053446  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
11025  053450  006001                          ROR     R1
11026  053452  006000                          ROR     R0
11027  053454  006001                          ROR     R1
11028  053456  006000                          ROR     R0
11029  053460  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
11030  053462  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
11031  053466  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
11032  053470  000016                  $OCTVL: .BLKB   14.              ;;RESERVE DATA TABLE
11033                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
11034
11035                                  ;;**********************************************************************
11036                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
11037                                  ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
11038                                  ;*POSITIVE.
11039                                  ;*CALL
11040                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
11041                                  ;*      JSR     PC,@#$DB2D
```

```
11042                                   ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
11043                                                                   ;;IS ON THE STACK
11044
11045
11046  053506  104407                   $DB2D:  SAVREG                  ;;SAVE REGISTERS
11047  053510  016602  000002                   MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
11048  053514  012700  053666                   MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
11049  053520  010066  000002                   MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
11050  053524  012201                           MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
11051  053526  012202                           MOV     (R2)+,R2
11052  053530  012737  000012  053604           MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
11053  053536  012704  053616                   MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
11054  053542  012705  053620                   MOV     #$TNPWR+2,R5
11055  053546  005003                   1$:     CLR     R3              ;;CLEAR PARTIAL
11056  053550  161401                   2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
11057  053552  005602                           SBC     R2
11058  053554  161502                           SUB     (R5),R2
11059  053556  002402                           BLT     3$              ;;BR IF TEN POWER TO LARGE
11060  053560  005203                           INC     R3              ;;ADD 1 TO PARTIAL
11061  053562  000772                           BR      2$              ;;LOOP
11062  053564  062401                   3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
11063  053566  005502                           ADC     R2
11064  053570  062402                           ADD     (R4)+,R2
11065  053572  022525                           CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
11066  053574  052703  000060                   BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
11067  053600  110320                           MOVB    R3,(R0)+        ;;SAVE IT
11068  053602  005327                           DEC     (PC)+           ;;DONE?
11069  053604  000000                   4$:     .WORD   0
11070  053606  001357                           BNE     1$              ;;BR IF NO
11071  053610  105020                           CLRB    (R0)+           ;;TERMINATOR
11072  053612  104410                           RESREG                  ;;RESTORE REGISTERS
11073  053614  000207                           RTS     PC              ;;RETURN
11074  053616  145000                   $TNPWR: 145000                  ;;1.0E09
11075  053620  035632                           35632
11076  053622  160400                           160400                  ;;1.0E08
11077  053624  002765                           2765
11078  053626  113200                           113200                  ;;1.0E07
11079  053630  000230                           230
11080  053632  041100                           041100                  ;;1.0E06
11081  053634  000017                           17
11082  053636  103240                           103240                  ;;1.0E05
11083  053640  000001                           1
11084  053642  023420                           23420                   ;;1.0E04
11085  053644  000000                           0
11086  053646  001750                           1750                    ;;1.0E03
11087  053650  000000                           0
11088  053652  000144                           144                     ;;1.0E02
11089  053654  000000                           0
11090  053656  000012                           12                      ;;1.0E01
11091  053660  000000                           0
11092  053662  000001                           1                       ;;1.0E00
11093  053664  000000                           0
11094  053666  000014                   $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
11095                                   .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
11096
11097                                   ;;**************************************************************
```

```
11098                                   ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
11099                                   ;*LEADING NUMBERS.
11100                                   ;*CALL
11101                                   ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
11102                                   ;*      JSR     PC,@#$SUPRS
11103
11104
11105  053702  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
11106  053704  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
11107  053710  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
11108  053712  001403                          BEQ     2$              ;;BR IF YES
11109  053714  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
11110  053720  001773                          BEQ     1$              ;;BR IF YES
11111  053722  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
11112  053724  010037  053732                  MOV     R0,3$           ;;SAVE FOR TYPING
11113  053730  104401                          TYPE                    ;;GO TYPE
11114  053732  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
11115  053734  012600                          MOV     (SP)+,R0        ;;RESTORE R0
11116  053736  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
11117  053740  000207                          RTS     PC              ;;RETURN
11118                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
11119
11120                                   ;;*****************************************************************
11121                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
11122                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
11123                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
11124                                   ;*CALL:
11125                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11126                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
11127                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
11128                                   ;*      .BYTE   M               ;;M=1 OR 0
11129                                   ;*                                      ;;1=TYPE LEADING ZEROS
11130                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
11131                                   ;*
11132                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
11133                                   ;*$TYPOS OR $TYPOC
11134                                   ;*CALL:
11135                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11136                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
11137                                   ;*
11138                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
11139                                   ;*CALL:
11140                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
11141                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
11.42
11143  053742  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
11144  053746  116637  000001  054165          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
11145  053754  112637  054167                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
11146  053760  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
11147  053764  000406                          BR      $TYPON
11148  053766  112737  000001  054165  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
11149  053774  112737  000006  054167          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
11150  054002  112737  000005  054164  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
11151  054010  010346                          MOV     R3,-(SP)        ;;SAVE R3
11152  054012  010446                          MOV     R4,-(SP)        ;;SAVE R4
11153  054014  010546                          MOV     R5,-(SP)        ;;SAVE R5
```

```
11154  054016  113704  054167            MOVB    $OMODE+1,R4     .;GET THE NUMBER OF DIGITS TO TYPE
11155  054022  005404                    NEG     R4
11156  054024  062704  000006            ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
11157  054030  110437  054166            MOVB    R4,$OMODE       ;;SAVE IT FOR JSE
11158  054034  113704  054165            MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
11159  054040  016605  000012            MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
11160  054044  005003                    CLR     R3              ;;CLEAR THE OUTPUT WORD
11161  054046  006105            1$:     ROL     R5              ;;ROTATE MSB INTO "C"
11162  054050  000404                    BR      3$              ;;GO DO MSB
11163  054052  006105            2$:     ROL     R5              ;;FORM THIS DIGIT
11164  054054  006105                    ROL     R5
11165  054056  006105                    ROL     R5
11166  054060  010503                    MOV     R5,R3
11167  054062  006103            3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
11168  054064  105337  054166            DECB    $OMODE          ;;TYPE THIS DIGIT?
11169  054070  100016                    BPL     7$              ;;BR IF NO
11170  054072  042703  177770            BIC     #177770,R3      ;;GET RID OF JUNK
11171  054076  001002                    BNE     4$              ;;TEST FOR 0
11172  054100  005704                    TST     R4              ;;SUPPRESS THIS 0?
11173  054102  001403                    BEQ     5$              ;;BR IF YES
11174  054104  005204            4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
11175  054106  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
11176  054112  052703  000040    5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
11177  054116  110337  054162            MOVB    R3,8$           ;;SAVE FOR TYPING
11178  054122  104401  054162            TYPE    ,8$             ;;GO TYPE THIS DIGIT
11179  054126  105337  054164    7$:     DECB    $OCNT           ;;COUNT BY 1
11180  054132  003347                    BGT     2$              ;;BR IF MORE TO DO
11181  054134  002402                    BLT     6$              ;;BR IF DONE
11182  054136  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
11183  054140  000744                    BR      2$              ;;GO DO THE LAST DIGIT
11184  054142  012605            6$:     MOV     (SP)+,R5        ;;RESTORE R5
11185  054144  012604                    MOV     (SP)+,R4        ;;RESTORE R4
11186  054146  012603                    MOV     (SP)+,R3        ;;RESTORE R3
11187  054150  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
11188  054156  012616                    MOV     (SP)+,(SP)
11189  054160  000002                    RTI                     ;;RETURN
11190  054162     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
11191  054163     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
11192  054164     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
11193  054165     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
11194  054166  000000            $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
11195                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
11196
11197                                  ;;**************************************************************
11198                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
11199                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
11200                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
11201                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
11202                                  ;*REPLACED WITH SPACES.
11203                                  ;*CALL:
11204                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
11205                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
11206
11207  054170                          $TYPDS:
11208  054170  010046                    MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11209  054172  010146                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
```

```
11210  054174  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11211  054176  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11212  054200  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11213  054202  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
11214  054206  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
11215  054212  100004                        BPL     1$              ;;BR IF INPUT IS POS.
11216  054214  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
11217  054216  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
11218  054224  005000                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
11219  054226  012703  054404                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
11220  054232  112723  000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
11221  054236  005002                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
11222  054240  016001  054374                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
11223  054244  160105                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
11224  054246  002402                        BLT     4$              ;;BR IF DONE
11225  054250  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
11226  054252  000774                        BR      3$
11227  054254  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
11228  054256  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
11229  054260  001002                        BNE     5$              ;;FALL THROUGH IF 0
11230  054262  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
11231  054264  100407                        BMI     7$              ;;BR IF YES
11232  054266  106316                5$:     ASLB    (SP)            ;;MSD?
11233  054270  103003                        BCC     6$              ;;BR IF NO
11234  054272  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
11235  054300  052702  000060          6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
11236  054304  052702  000040          7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
11237  054310  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
11238  054312  005720                        TST     (R0)+           ;;JUST INCREMENTING
11239  054314  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
11240  054320  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
11241  054322  003002                        BGT     8$              ;;GO TO EXIT
11242  054324  010502                        MOV     R5,R2           ;;GET THE LSD
11243  054326  000764                        BR      6$              ;;GO CHANGE TO ASCII
11244  054330  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
11245  054332  100003                        BPL     9$              ;;BR IF NO
11246  054334  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
11247  054342  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
11248  054344  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
11249  054346  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
11250  054350  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
11251  054352  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
11252  054354  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
11253  054356  104401  054404                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
11254  054362  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
11255  054370  012616                        MOV     (SP)+,(SP)
11256  054372  000002                        RTI                     ;;RETURN TO USER
11257  054374  023420                $DTBL:  10000.
11258  054376  001750                        1000.
11259  054400  000144                        100.
11260  054402  000012                        10.
11261  054404  000004                $DBLK:  .BLKW   4
11262                                 .SBTTL  ERROR HANDLER ROUTINE
11263
11264                                 ;;**********************************************************************
11265                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

```
11266                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
11267                                   ;*AND GO TO TYPERR ON ERROR
11268                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11269                                   ;*SW15=1        HALT ON ERROR
11270                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
11271                                   ;*SW10=1        BELL ON ERROR
11272                                   ;*SW09=1        LOOP ON ERROR
11273                                   ;*CALL
11274                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
11275
11276  054414                           $ERROR:
11277  054414  105237  001103           7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
11278  054420  001775                           BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
11279  054422  013777  001102  124512           MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
11280  054430  032777  002000  124502           BIT     #BIT10,@SWR     ;;BELL ON ERROR?
11281  054436  001402                           BEQ     1$              ;;NO - SKIP
11282  054440  104401  001310                   TYPE    ,$BELL          ;;RING BELL
11283  054444  005237  001112           1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
11284  054450  011637  001116                   MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
11285  054454  162737  000002  001116           SUB     #2,$ERRPC
11286  054462  117737  124430  001114           MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
11287  054470  032777  020000  124442           BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
11288  054476  001004                           BNE     20$             ;;SKIP TYPEOUTS
11289  054500  004737  040732                   JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
11290  054504  104401  001315                   TYPE    ,$CRLF
11291  054510                           20$:
11292  054510  122737  000001  001340           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
11293  054516  001007                           BNE     2$              ;;NO,SKIP APT ERROR REPORT
11294  054520  113737  001114  054532           MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
11295  054526  004737  055374                   JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
11296  054532     000           21$:    .BYTE   0
11297  054533     000                   .BYTE   0
11298  054534  000777           22$:    BR      22$             ;;APT ERROR LOOP
11299  054536  005777  124376   2$:     TST     @SWR            ;;HALT ON ERROR
11300  054542  100001                   BPL     3$              ;;SKIP IF CONTINUE
11301  054544  000000                   HALT                    ;;HALT ON ERROR!
11302  054546  032777  001000  124364   3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
11303  054554  001402                   BEQ     4$              ;;BR IF NO
11304  054556  013716  001110                   MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
11305  054562  005737  001306           4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
11306  054566  001402                   BEQ     5$              ;;BR IF NONE
11307  054570  013716  001306                   MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
11308  054574                           5$:
11309  054574  022737  025462  000042           CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
11310  054602  001001                   BNE     6$              ;;BRANCH IF NO
11311  054604  000000                   HALT                    ;;YES
11312  054606                           6$:
11313  054606  000002                   RTI                     ;;RETURN
11314                                   .SBTTL  TTY INPUT ROUTINE
11315
11316                                   ;;**************************************************************************
11317                                   .ENABL  LSB
11318
11319                                   .DSABL  LSB
11320
11321
```

```
11322                                   ;;*****************************************************************
11323                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
11324                                   ;*CALL:
11325                                   ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
11326                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
11327                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
11328                                   ;
11329
11330  054610  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
11331  054612  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
11332  054620  105777  124320          1$:     TSTB    @$TKS           ;;WAIT FOR
11333  054624  100375                          BPL     1$              ;;A CHARACTER
11334  054626  117766  124314  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
11335  054634  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
11336  054642  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
11337  054650  001013                          BNE     3$              ;;BRANCH IF NO
11338  054652  105777  124266          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
11339  054656  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
11340  054660  117746  124262                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
11341  054664  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
11342  054670  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
11343  054674  001366                          BNE     2$              ;;IF NOT DISCARD IT
11344  054676  000750                          BR      1$              ;;YES, RESUME
11345  054700  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
11346  054706  002407                          BLT     4$              ;;BRANCH IF YES
11347  054710  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
11348  054716  003003                          BGT     4$              ;;BRANCH IF YES
11349  054720  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
11350  054726  000002                  4$:     RTI                     ;;GO BACK TO USER
11351  054730  052536  005015     000  $CNTLU: .ASCIZ   /^U/<15><12>    ;;CONTROL "U"
11352  054735     136  006507  000012  $CNTLG: .ASCIZ   /^G/<15><12>    ;;CONTROL "G"
11353  054742  005015  053523  020122  $MSWR:  .ASCIZ   <15><12>/SWR = /
11354  054750  020075     000
11355  054753     040  047040  053505  $MNEW:  .ASCIZ   /  NEW = /
11356  054760  036440  000040
11357
11358
11359
11360                                   .SBTTL  POWER DOWN AND UP ROUTINES
11361
11362                                   ;POWER DOWN ROUTINE
11363  054764  012737  054776  000024  $PWRDN: MOV     #$PWRUP,PWRVEC  ;SET VECTOR FOR POWER UP
11364  054772  000000                          HALT                    ;HANG UP
11365  054774  000776                          BR      .-2
11366
11367                                   ;POWER UP ROUTINE
11368  054776  005037  055050          $PWRUP: CLR     $PWRCT          ;WAIT LOOP FOR TTY TO COME UP
11369  055002  005237  055050          4$:     INC     $PWRCT
11370  055006  001375                          BNE     4$
11371  055010  012737  054764  000024          MOV     #$PWRDN,PWRVEC  ;SET VECTOR FOR POWER DOWN
11372  055016  012737  000340  000026          MOV     #PR7,PWRVEC+2   ;RE-ESTABLISH POWER AND TRAP PRIORITIES
11373  055024  012737  000340  000036          MOV     #PR7,TRAPVEC+2
11374  055032  012706  001100                  MOV     #STACK,SP       ;RE-INITIALIZE THE STACK
11375  055036  104401  055052                  TYPE    ,PWRMSG         ;TYPE "POWER FAILED"
11376  055042  000005                          RESET                   ;CLEAR THE UNIBUS
11377  055044  000177  124036                  JMP     @$LPADR         ;RESTART THE CURRENT TEST
```

```
11378  055050  000000                  $PWRCT: .WORD   0
11379  055052  005015  047520  042527  PWRMSG: .ASCIZ  <15><12>/POWER FAILED/<15><12>
11380  055060  020122  040506  046111
11381  055066  042105  005015     000
11382          055074                  .EVEN
11383
11384
11385
11386                                  ;;*****************************************************************
11387                                  ;* SCOPE1 - INTERNAL SCOPE ON ERROR ROUTINE
11388                                  ;*CALLED BY "SCOPER"
11389                                  ;;*****************************************************************
11390  055074  105737  001103          SCOPE1: TSTB    $ERFLG          ;SEE IF AN ERROR HAS OCCURRED
11391  055100  001406                          BEQ     6$              ;BR IF NOT
11392  055102  032777  001000  124030          BIT     #BIT9,@SWR      ;SEE IF LOOP ON ERROR DESIRED
11393  055110  001402                          BEQ     6$              ;BR IF NOT
11394  055112  013716  001110                  MOV     $LPERR,(SP)     ;SET ERROR LOOP ADDRESS ON STACK
11395  055116  000002                  6$:     RTI                     ;RETURN
11396
11397
11398
11399                                  .SBTTL  SCOPE HANDLER ROUTINE
11400
11401                                  ;;*****************************************************************
11402                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
11403                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
11404                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
11405                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
11406                                  ;*SW14=1         LOOP ON TEST
11407                                  ;*SW11=1         INHIBIT ITERATIONS
11408                                  ;*SW09=1         LOOP ON ERROR
11409                                  ;*CALL
11410                                  ;*      SCOPE           ;;SCOPE=IOT
11411
11412  055120                          $SCOPE:
11413  055120  005737  001304                  TST     $TIMES          ;CHECK CURRENT ITERATION NUMBER
11414  055124  001404                          BEQ     $XTSTR          ;BR IF 0, TO SKIP CHECK OF SWR BIT 14
11415  055126  032777  040000  124004  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
11416  055134  001101                          BNE     $OVER           ;;YES IF SW14=1
11417                                  ;#####START OF CODE FOR THE XOR TESTER#####
11418  055136  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
11419                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
11420  055140  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
11421  055144  012737  055164  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
11422  055152  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
11423  055156  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11424  055162  000450                          BR      $SVLAD          ;;GO TO THE NEXT TEST
11425  055164  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
11426  055166  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
11427  055172  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
11428  055174                          6$:;#####END OF CODE FOR THE XOR TESTER#####
11429  055174  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
11430  055200  001421                          BEQ     3$              ;;BR IF NO
11431  055202  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
11432  055210  101015                          BHI     3$              ;;BR IF NO
11433  055212  032777  001000  123720          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
```

```
11434  055220  001404                          BEQ     4$              ;;BR IF NO
11435  055222  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
11436  055230  000443                          BR      $OVER
11437  055232  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
11438  055236  005037  001304                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
11439  055242  000412                          BR      1$              ;;ESCAPE TO THE NEXT TEST
11440  055244  032777  004000  123666  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
11441  055252  001006                          BNE     1$              ;;BR IF YES
11442  055254  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
11443  055260  023737  001304  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
11444  055266  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
11445  055270  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
11446  055276  013737  055354  001304          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
11447  055304  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
11448  055310  113737  001102  001324          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
11449  055316  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
11450  055322  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
11451  055326  005037  001306                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
11452  055332  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
11453  055340  013777  001102  123574  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
11454  055346  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
11455  055352  000002                          RTI                     ;;FIXES PS
11456  055354  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
11457                                   .SBTTL  APT COMMUNICATIONS ROUTINE
11458
11459                                   ;;**************************************************************************
11460  055356  112737  000001  055622  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
11461  055364  112737  000001  055620  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
11462  055372  000403                          BR      $ATYC
11463  055374  112737  000001  055622  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
11464  055402                          $ATYC:
11465  055402  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11466  055404  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11467  055406  105737  055620                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
11468  055412  001450                          BEQ     5$              ;;IF NOT:  BR
11469  055414  122737  000001  001340          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
11470  055422  001031                          BNE     3$              ;;IF NOT:  BR
11471  055424  132737  000100  001341          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
11472  055432  001425                          BEQ     3$              ;;IF NOT:  BR
11473  055434  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
11474  055440  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
11475  055446  005737  001320          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
11476  055452  001375                          BNE     1$              ;;IF NOT:  WAIT
11477  055454  010037  001334                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
11478  055460  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
11479  055462  001376                          BNE     2$
11480  055464  163700  001334                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
11481  055470  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
11482  055472  010037  001336                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
11483  055476  012737  000004  001320          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
11484  055504  000413                          BR      5$
11485  055506  017637  000004  055532  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
11486  055514  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
11487  055522  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
11488  055526  004737  052666                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
11489  055532  000000                  4$:     .WORD   0
```

```
11490  055534                          5$:
11491  055534  105737  055622          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
11492  055540  001416                          BEQ     12$             ;;IF NOT:  BR
11493  055542  005737  001340                  TST     $ENV            ;;RUNNING UNDER APT?
11494  055546  001413                          BEQ     12$             ;;IF NOT:  BR
11495  055550  005737  001320          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
11496  055554  001375                          BNE     11$             ;;IF NOT:  WAIT
11497  055556  017637  000004  001322          MOV     @4(SP),$FATAL   ;;GET ERROR #
11498  055564  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
11499  055572  005237  001320                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
11500  055576  105037  055622          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
11501  055602  105037  055621                  CLRB    $LFLG           ;;CLEAR LOG FLAG
11502  055606  105037  055620                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
11503  055612  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11504  055614  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11505  055616  000207                          RTS     PC              ;;RETURN
11506  055620     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
11507  055621     000          $LFLG:  .BYTE   0               ;;LOG FLAG
11508  055622     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
11509             055624                  .EVEN
11510             000200          APTSIZE=200
11511             000001          APTENV=001
11512             000100          APTSPOOL=100
11513             000040          APTCSUP=040
11514                             .SBTTL  ROUTINE TO SIZE MEMORY
11515
11516                             ;;**************************************************************************
11517                             ;*CALL:
11518                             ;*      JSR     PC,$SIZE
11519                             ;*      RETURN
11520                             ;*$LSTAD WILL CONTAIN:
11521                             ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
11522                             ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
11523                             ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
11524                             ;*$KT11 IS THE MEMORY MANAGEMENT KEY
11525                             ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
11526                             ;*      MUST BE SETUP BEFORE THE CALL
11527                             ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
11528                             ;*      DETERMINED BY ROUTINE
11529
11530  055624  010046          $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
11531  055626  010146                  MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
11532  055630  010246                  MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
11533  055632  010346                  MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
11534  055634  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
11535  055640  013746  000006          MOV     @#ERRVEC+2,-(SP)
11536  055644  010600                  MOV     SP,R0           ;;SAVE THE STACK POINTER
11537                          ;;SET THE ERRVEC PS TO THE PRESENT PS
11538  055646  104400                  TRAP                    ;;PUSH OLD PSW AND PC ON STACK
11539  055650  012637  000006          MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
11540  055654  012701  003776          MOV     #3776,R1        ;;SETUP ADDRESS
11541  055660  105727                  TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
11542  055662  000200          $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
11543  055664  100062                  BPL     $CORE           ;;BR IF NO
11544  055666  012737  056024  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
11545  055674  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
```

```
11546  055700  052737  100000  055662          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
11547  055706  005046                          CLR     -(SP)           ;;INITIALIZE FOR 'PAR' LOADING
11548  055710  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
11549  055714  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT 'PAR.'S AND EIGHT 'PDR.'S
11550  055720  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
11551  055724  011622                          MOV     (SP),(R2)+      ;;LOAD 'PAR'
11552  055730  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT 'PAR'
11553  055734  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
11554  055736  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
11555  055742  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
11556  055744  012737  055762  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
11557  055752  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
11558  055760  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
11559  055762  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
11560  055764  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
11561  055770  012737  056014  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
11562  055776  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
11563  056002  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
11564  056006  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
11565  056012  101371                          BHI     4$              ;;NO--TRY IT
11566  056014  011202                  $XTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
11567  056016  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
11568  056022  000421                          BR      $SIZEX
11569  056024  042737  100000  055662  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
11570  056032  012737  056062  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
11571  056040  005002                          CLR     R2              ;;SET UP BANK
11572  056042  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
11573  056046  062702  000040                  ADD     #40,R2          ;;1K STEP
11574  056052  005711                          TST     (R1)            ;;TRAP ON TIME OUT
11575  056054  022701  177776                  CMP     #177776,R1      ;;LAST ONE
11576  056060  001370                          BNE     1$              ;;NO--TRY AGAIN
11577  056062  162701  004000  $CROUT: SUB     #4000,R1
11578  056066  162702  000040  $SIZEX: SUB     #40,R2          ;;DROP BACK
11579  056072  010006                  MOV     R0,SP           ;;RESTORE THE STACK
11580  056074  012637  000006          MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
11581  056100  012637  000004          MOV     (SP)+,@#ERRVEC
11582  056104  010137  056126          MOV     R1,$LSTAD       ;;LAST ADDRESS
11583  056110  010237  056130          MOV     R2,$LSTBK       ;;LAST BANK
11584  056114  012603                  MOV     (SP)+,R3        ;;RESTORE R3
11585  056116  012602                  MOV     (SP)+,R2        ;;RESTORE R2
11586  056120  012601                  MOV     (SP)+,R1        ;;RESTORE R1
11587  056122  012600                  MOV     (SP)+,R0        ;;RESTORE R0
11588  056124  000207                  RTS     PC
11589  056126  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
11590  056130  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
11591                          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
11592
11593                          ;;*****************************************************************
11594                          ;*SAVE R0-R5
11595                          ;*CALL:
11596                          ;*      SAVREG
11597                          ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
11598                          ;*
11599                          ;*TOP---(+16)
11600                          ;* +2---(+18)
11601                          ;* +4---R5
```

```
11602                                   ;* +6---R4
11603                                   ;* +8---R3
11604                                   ;*+10---R2
11605                                   ;*+12---R1
11606                                   ;*+14---R0
11607
11608  056132                           $SAVREG:
11609  056132  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
11610  056134  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
11611  056136  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
11612  056140  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
11613  056142  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
11614  056144  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
11615  056146  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
11616  056152  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
11617  056156  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
11618  056162  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
11619  056166  000002                          RTI
11620
11621                                   ;*RESTORE R0-R5
11622                                   ;*CALL:
11623                                   ;*      RESREG
11624  056170                           $RESREG:
11625  056170  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
11626  056174  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
11627  056200  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
11628  056204  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
11629  056210  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
11630  056212  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
11631  056214  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
11632  056216  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
11633  056220  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
11634  056222  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
11635  056224  000002                          RTI
11636                                   .SBTTL  TRAP DECODER
11637
11638                                   ;;**************************************************************************
11639                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
11640                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
11641                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
11642                                   ;*GO TO THAT ROUTINE.
11643
11644  056226  010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
11645  056230  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
11646  056234  005740                          TST     -(R0)           ;;BACKUP BY 2
11647  056236  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
11648  056240  006300                          ASL     R0              ;;POSITION FOR INDEXING
11649  056242  016000  056262                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
11650  056246  000200                          RTS     R0              ;;GO TO ROUTINE
11651
11652
11653                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
11654
11655  056250  011646                   $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
11656  056252  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
11657  056260  000002                          RTI                     ;;RESTORE THE PSW
```

```
11658
11659                                   .SBTTL  TRAP TABLE
11660
11661                                   ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
11662                                   ;*BY THE "TRAP" INSTRUCTION.
11663
11664                                   ;       ROUTINE
11665                                   ;       -------
11666  056262  056250                  $TRPAD: .WORD   $TRAP2
11667  056264  052666                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
11668  056266  053766                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
11669  056270  053742                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
11670  056272  054002                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
11671  056274  054170                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
11672
11673
11674  056276  054610                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
11675  056300  056132                          $SAVREG ;;CALL=SAVREG   TRAP+7(104407)  SAVE R0-R5 ROUTINE
11676  056302  056170                          $RESREG ;;CALL=RESREG   TRAP+10(104410) RESTORE R0-R5 ROUTINE
11677  056304  055074                          SCOPE1  ;;CALL=SCOPER   TRAP+11(104411) INTERNAL LOOP ON ERROR ROUTINE
11678
11679  056306  020040  020040  042524  TSTMSG: .ASCIZ  /    TEST /
11680  056314  052123  000040
11681  056320  025052  020040     000  AS2SP2: .ASCIZ  /**  /
11682  056325     125  044516  052502  EM1:    .ASCIZ  /UNIBUS PAR ERR/
11683  056332  020123  040520  020122
11684  056340  051105  000122
11685  056344  047516  026516  054105  EM2:    .ASCIZ  /NON-EXIST MEM/
11686  056352  051511  020124  042515
11687  056360  000115
11688  056362  047516  026516  054105  EM3:    .ASCIZ  /NON-EXIST DRV/
11689  056370  051511  020124  051104
11690  056376  000126
11691  056400  047125  052111  043040  EM4:    .ASCIZ  /UNIT FIELD ERR/
11692  056406  042511  042114  042440
11693  056414  051122     000
11694  056417     123  041125  054523  EM5:    .ASCIZ  /SUBSYS TIMEOUT/
11695  056424  020123  044524  042515
11696  056432  052517  000124
11697  056436  020104  047524  041440  EM6:    .ASCIZ  /D TO C PAR ERR/
11698  056444  050040  051101  042440
11699  056452  051122     000
11700  056455     103  052040  020117  EM7:    .ASCIZ  /C TO D PAR ERR/
11701  056462  020104  040520  020122
11702  056470  051105  000122
11703  056474  041501  046040  053517  EM10:   .ASCIZ  /AC LOW/
11704  056502     000
11705  056503     123  042520  042105  EM11:   .ASCIZ  /SPEED LOSS/
11706  056510  046040  051517  000123
11707  056516  046111  020114  052506  EM12:   .ASCIZ  /ILL FUNCT/
11708  056524  041516  000124
11709  056530  043520  020115  051105  EM13:   .ASCIZ  /PGM ERR/
11710  056536  000122
11711  056540  047516  026516  054105  EM14:   .ASCIZ  /NON-EXIST FUNCT/
11712  056546  051511  020124  052506
11713  056554  041516  000124
```

```
11714  056560  051104  020126  054524  EM15:   .ASCIZ  /DRV TYP ERR/
11715  056566  020120  051105  000122
11716  056574  046506  020124  051105  EM16:   .ASCIZ  /FMT ERR/
11717  056602  000122
11718  056604  051127  020124  047514  EM17:   .ASCIZ  /WRT LOCK ERR/
11719  056612  045503  042440  051122
11720  056620     000
11721  056621     104  053122  052440  EM20:   .ASCIZ  /DRV UNSAFE/
11722  056626  051516  043101  000105
11723  056634  042523  045505  044440  EM21:   .ASCIZ  /SEEK INC/
11724  056642  041516     000
11725  056645     103  046131  047440  EM22:   .ASCIZ  /CYL OVRFLO/
11726  056652  051126  046106  000117
11727  056660  046111  020114  054503  EM23:   .ASCIZ  /ILL CYL ADDR/
11728  056666  020114  042101  051104
11729  056674     000
11730  056675     104  053122  047440  EM24:   .ASCIZ  /DRV OFF TRK/
11731  056702  043106  052040  045522
11732  056710     000
11733  056711     104  053122  052040  EM25:   .ASCIZ  /DRV TIMING ERR/
11734  056716  046511  047111  020107
11735  056724  051105  000122
11736  056730  040504  040524  046040  EM26:   .ASCIZ  /DATA LATE/
11737  056736  052101  000105
11738  056742  047503  052116  052040  EM27:   .ASCIZ  /CONT TIMEOUT/
11739  056750  046511  047505  052125
11740  056756     000
11741  056757     117  020120  047111  EM30:   .ASCIZ  /OP INCOMP/
11742  056764  047503  050115     000
11743  056771     110  051104  053040  EM31:   .ASCIZ  /HDR VRC ERR/
11744  056776  041522  042440  051122
11745  057004     000
11746  057005     104  052101  020101  EM32:   .ASCIZ  /DATA CHK ERR/
11747  057012  044103  020113  051105
11748  057020  000122
11749  057022  051127  020124  044103  EM33:   .ASCIZ  /WRT CHK ERR/
11750  057030  020113  051105  000122
11751  057036  040504  040524  046440  EM34:   .ASCIZ  /DATA MISCOMPARE/
11752  057044  051511  047503  050115
11753  057052  051101  000105
11754  057056  047516  042040  053122  EM35:   .ASCIZ  /NO DRV RESPONSE-UFE & NXD/
11755  057064  051040  051505  047520
11756  057072  051516  026505  043125
11757  057100  020105  020046  054116
11758  057106  000104
11759  057110  051104  020126  051105  EM36:   .ASCIZ  /DRV ERR WILL NOT CLEAR/
11760  057116  020122  044527  046114
11761  057124  047040  052117  041440
11762  057132  042514  051101     000
11763  057137     104  053122  051440  EM37:   .ASCIZ  /DRV STATUS CHANGE WILL NOT CLEAR/
11764  057144  040524  052524  020123
11765  057152  044103  047101  042507
11766  057160  053440  046111  020114
11767  057166  047516  020124  046103
11768  057174  040505  000122
11769  057200  052101  023524  020116  EM40:   .ASCIZ  /ATT'N BUT NO STATUS CHANGE OR FAULT/
```

```
11770  057206  052502  020124  047516
11771  057214  051440  040524  052524
11772  057222  020123  044103  047101
11773  057230  042507  047440  020122
11774  057236  040506  046125  000124
11775  057244  052101  023524  020116  EM41:   .ASCIZ  /ATT'N BUT DRV NOT AVAIL/
11776  057252  052502  020124  051104
11777  057260  020126  047516  020124
11778  057266  053101  044501  000114
11779  057274  052101  023524  020116  EM42:   .ASCIZ  /ATT'N WHEN NOT EXPECTED/
11780  057302  044127  047105  047040
11781  057310  052117  042440  050130
11782  057316  041505  042524  000104
11783  057324  051105  020122  040507  EM43:   .ASCIZ  /ERR GATHERING DRV STATUS/
11784  057332  044124  051105  047111
11785  057340  020107  051104  020126
11786  057346  052123  052101  051525
11787  057354     000
11788  057355     115  046125  020124  EM52:   .ASCIZ  /MULT DRV SEL/
11789  057362  051104  020126  042523
11790  057370  000114
11791  057372  042110  020122  047503  EM53:   .ASCIZ  /HDR COMP ERR/
11792  057400  050115  042440  051122
11793  057406     000
11794  057407     123  041125  054523  EM56:   .ASCIZ  /SUBSYS TIMEOUT/
11795  057414  020123  044524  042515
11796  057422  052517  000124
11797  057426  051105  020122  047111  EM60:   .ASCIZ  /ERR IN RECAL FOR RECOVERY/
11798  057434  051040  041505  046101
11799  057442  043040  051117  051040
11800  057450  041505  053117  051105
11801  057456  000131
11802  057460  051104  050117  044520  EM61:   .ASCIZ  /DROPPING DRV, FATAL ERR/
11803  057466  043516  042040  053122
11804  057474  020054  040506  040524
11805  057502  020114  051105  000122
11806  057510  054503  020114  044515  EM62:   .ASCIZ  /CYL MISCOMP/
11807  057516  041523  046517  000120
11808  057524  046103  040505  020122  EM63:   .ASCIZ  /CLEAR CONTR DID NOT CLEAR ERR/
11809  057532  047503  052116  020122
11810  057540  044504  020104  047516
11811  057546  020124  046103  040505
11812  057554  020122  051105  000122
11813  057562  047516  040440  052124  EM64:   .ASCIZ  /NO ATT'N IN ATT'N REG/
11814  057570  047047  044440  020116
11815  057576  052101  023524  020116
11816  057604  042522  000107
11817  057610  047125  047523  044514  EM65:   .ASCIZ  /UNSOLICITED ATT'N/
11818  057616  044503  042524  020104
11819  057624  052101  023524  000116
11820  057632  047125  054105  020120  EM66:   .ASCIZ  /UNEXP DATA ERR/
11821  057640  040504  040524  042440
11822  057646  051122     000
11823  057651     101  052124  047047  EM67:   .ASCIZ  /ATT'N DID NOT RESET WITH CLEAR/
11824  057656  042040  042111  047040
11825  057664  052117  051040  051505
```

```
11826  057672  052105  053440  052111
11827  057700  020110  046103  040505
11828  057706  000122
11829  057710  052523  051502  051531  EM70:   .ASCIZ  /SUBSYS CLEAR DID NOT CLEAR DRV ATT'N/
11830  057716  041440  042514  051101
11831  057724  042040  042111  047040
11832  057732  052117  041440  042514
11833  057740  051101  042040  053122
11834  057746  040440  052124  047047
11835  057754     000
11836  057755     104  052101  020101  EM71:   .ASCIZ  /DATA LATE WHEN READING HDR/
11837  057762  040514  042524  053440
11838  057770  042510  020116  042522
11839  057776  042101  047111  020107
11840  060004  042110  000122
11841  060010  047503  052116  020122  EM72:   .ASCIZ  /CONTR ERR WHEN DRIVER SERV/
11842  060016  051105  020122  044127
11843  060024  047105  042040  044522
11844  060032  042526  020122  042523
11845  060040  053122     000
11846  060043     104  053122  042040  EM73:   .ASCIZ  /DRV DET PAR ERR/
11847  060050  052105  050040  051101
11848  060056  042440  051122     000
11849  060063     125  042116  043105  EM74:   .ASCIZ  /UNDEF ERR/
11850  060070  042440  051122     000
11851  060075     115  051101  044513  EM75:   .ASCIZ  /MARKING THIS SECTOR BAD/
11852  060102  043516  052040  044510
11853  060110  020123  042523  052103
11854  060116  051117  041040  042101
11855  060124     000
11856  060125     102  042101  042040  EM76:   .ASCIZ  /BAD DATA VERIF'N WITH READ. ECC OF LAST RETRY IS:/
11857  060132  052101  020101  042526
11858  060140  044522  023506  020116
11859  060146  044527  044124  051040
11860  060154  040505  027104  042440
11861  060162  041503  047440  020106
11862  060170  040514  052123  051040
11863  060176  052105  054522  044440
11864  060204  035123     000
11865  060207     122  052105  054522  EM77:   .ASCIZ  /RETRY SUCCESSFUL/
11866  060214  051440  041525  042503
11867  060222  051523  052506  000114
11868  060230  042522  051124  020131  EM100:  .ASCIZ  /RETRY UNSUCCESSFUL/
11869  060236  047125  052523  041503
11870  060244  051505  043123  046125
11871  060252     000
11872  060253     103  047101  047516  EM101:  .ASCIZ  /CANNOT FIND A VALID HEADER IN TRACK JUST READ/
11873  060260  020124  044506  042116
11874  060266  040440  053040  046101
11875  060274  042111  044040  040505
11876  060302  042504  020122  047111
11877  060310  052040  040522  045503
11878  060316  045040  051525  020124
11879  060324  042522  042101     000
11880  060331     102  042101  051440  EM102:  .ASCIZ  /BAD SECT ERR ON SECT NOT LISTED BAD/
11881  060336  041505  020124  051105
```

```
11882  060344  020122  047117  051440
11883  060352  041505  020124  047516
11884  060360  020124  044514  052123
11885  060366  042105  041040  042101
11886  060374     000
11887  060375     124  046511  042105  EM103:  .ASCIZ  /TIMED-OUT ON READ HDR/
11888  060402  047455  052125  047440
11889  060410  020116  042522  042101
11890  060416  044040  051104     000
11891  060423     124  046511  042105  EM104:  .ASCIZ  /TIMED-OUT ON SEEK/
11892  060430  047455  052125  047440
11893  060436  020116  042523  045505
11894  060444     000
11895  060445     104  053122  051440  EM105:  .ASCIZ  /DRV SIEZED BY OTHER PORT/
11896  060452  042511  042532  020104
11897  060460  054502  047440  044124
11898  060466  051105  050040  051117
11899  060474  000124
11900  060476  040504  040524  046440  EM106:  .ASCIZ  /DATA MISCMPR WHILE BAI SET/
11901  060504  051511  046503  051120
11902  060512  053440  044510  042514
11903  060520  041040  044501  051440
11904  060526  052105     000
11905  060531     116  020117  042516  EM107:  .ASCIZ  /NO NEM ERR WHEN REF'ING LOC 760000/
11906  060536  020115  051105  020122
11907  060544  044127  047105  051040
11908  060552  043105  044447  043516
11909  060560  046040  041517  033440
11910  060566  030066  030060  000060
11911  060574  047111  051124  052120  EM110:  .ASCIZ  /INTRPT WHEN CNTRLR NOT RDY/
11912  060602  053440  042510  020116
11913  060610  047103  051124  051114
11914  060616  047040  052117  051040
11915  060624  054504     000
11916  060627     116  020117  052101  EM111:  .ASCIZ  /NO ATT'N ON SEEK/
11917  060634  023524  020116  047117
11918  060642  051440  042505  000113
11919  060650  051104  020126  054503  EM112:  .ASCIZ  /DRV CYL INCORRECT/
11920  060656  020114  047111  047503
11921  060664  051122  041505  000124
11922  060672  041101  051117  026524  EM113:  .ASCIZ  /ABORT- CAN'T READ BSF/
11923  060700  041440  047101  052047
11924  060706  051040  040505  020104
11925  060714  051502  000106
11926  060720  052113  030461  043040  EM114:  .ASCIZ  /KT11 FAILURE/
11927  060726  044501  052514  042522
11928  060734     000
11929  060735     115  046505  050040  EM115:  .ASCIZ  /MEM PAR ERR/
11930  060742  051101  042440  051122
11931  060750     000
11932  060751     103  051125  042522  DH100:  .ASCIZ  /CURRENT CMD :/
11933  060756  052116  041440  042115
11934  060764  035040     000
11935  060767     120  042522  020126  DH105:  .ASCIZ  /PREV CMD :/
11936  060774  046503  020104  000072
11937  061002  045522  030466  020061  DH200:  .ASCIZ  /RK611 REGS :/
```

```
11938  061010  042522  051507  035040
11939  061016     000
11940  061017     122  046505  044501  DH500:  .ASCIZ  /REMAINING REGS NOT VALID/
11941  061024  044516  043516  051040
11942  061032  043505  020123  047516
11943  061040  020124  040526  044514
11944  061046  000104
11945  061050  044124  020105  047506  DH501:  .ASCIZ  /THE FOLLOWING REG DATA MAY BE INCORRECT :/
11946  061056  046114  053517  047111
11947  061064  020107  042522  020107
11948  061072  040504  040524  046440
11949  061100  054501  041040  020105
11950  061106  047111  047503  051122
11951  061114  041505  020124  000072
11952  061122  051105  020122  041520  DH101:  .ASCIZ  /ERR PC  DRIVE   CMND    CYLNDR  TRACK   SECTOR  WD CNT/
11953  061130  020040  051104  053111
11954  061136  020105  020040  046503
11955  061144  042116  020040  020040
11956  061152  054503  047114  051104
11957  061160  020040  051124  041501
11958  061166  020113  020040  042523
11959  061174  052103  051117  020040
11960  061202  042127  041440  052116
11961  061210     000
11962  061211     110  020111  040502  DH102:  .ASCIZ  /HI BA   LO BA/
11963  061216  020040  046040  020117
11964  061224  040500     000
11965  061227     120  042522  027126  DH103:  .ASCIZ  /PREV. UNIBUS MAP :/
11966  061234  052440  044516  052502
11967  061242  020123  040515  020120
11968  061250  000072
11969  061252  044510  051040  043505  DH104:  .ASCIZ  /HI REG0 LO REG0/
11970  061260  020060  047514  051040
11971  061266  043505  000060
11972  061272  052503  027122  052440  DH106:  .ASCIZ  /CUR. UNIBUS MAP :/
11973  061300  044516  052502  020123
11974  061306  040515  020120  000072
11975  061314  045522  051503  020061  DH201:  .ASCIZ  /RKCS1   RKCS2   RKDC    RKDA    RKWCR   RKBA    RKASOF/
11976  061322  020040  045522  051503
11977  061330  020062  020040  045522
11978  061336  041504  020040  020040
11979  061344  045522  040504  020040
11980  061352  020040  045522  041527
11981  061360  020122  020040  045522
11982  061366  040502  020040  020040
11983  061374  045522  051501  043117
11984  061402     000
11985  061403     122  042113  020123  DH202:  .ASCIZ  /RKDS    RKER/
11986  061410  020040  051040  042513
11987  061416  000122
11988  061420  030101  020060  020040  DH203:  .ASCIZ  /A00     B00     A01     B01     A02     B02     A03     B03/
11989  061426  020040  030102  020060
11990  061434  020040  020040  030101
11991  061442  020061  020040  020040
11992  061450  030102  020061  020040
11993  061456  020040  030101  020062
```

```
 11994  061464  020040  020040  030102
 11995  061472  020062  020040  020040
 11996  061500  030101  020063  020040
 11997  061506  020040  030102  000063
 11998  061514  042110  020122  044123  DH601:  .ASCIZ  /HDR SHOULD BE:/
 11999  061522  052517  042114  041040
 12000  061530  035105     000
 12001  061533     110  051104  047440  DH602:  .ASCIZ  /HDR OF SECTOR 0 THIS CYL IS:/
 12002  061540  020106  042523  052103
 12003  061546  051117  030040  052040
 12004  061554  044510  020123  054503
 12005  061562  020114  051511  000072
 12006  061570  040520  045503  040440  DH604:  .ASCIZ  /PACK ADDR OF ERR(S) :/
 12007  061576  042104  020122  043117
 12008  061604  042440  051122  051450
 12009  061612  020051  000072
 12010  061616  054503  047114  051104  DH6041: .ASCIZ  /CYLNDR  TRACK   SECTOR/
 12011  061624  020040  051124  041501
 12012  061632  020113  020040  042523
 12013  061640  052103  051117     000
 12014  061645     103  046131  042116  DH6042: .ASCIZ  /CYLNDR  TRACK/
 12015  061652  020122  052040  040522
 12016  061660  045503     000
 12017  061663     105  041503  042040  DH605:  .ASCIZ  /ECC DATA IS:/
 12018  061670  052101  020101  051511
 12019  061676  000072
 12020  061700  047520  020123  020040  DH6051: .ASCIZ  /POS     PAT     CORRECTABLE?/
 12021  061706  020040  040520  020124
 12022  061714  020040  020040  047503
 12023  061722  051122  041505  040524
 12024  061730  046102  037505     000
 12025  061735     127  051117  030504  DH606:  .ASCIZ  /WORD1   WORD2   WORD3/
 12026  061742  020040  053440  051117
 12027  061750  031104  020040  053440
 12028  061756  051117  031504     000
 12029  061763     102  042101  044040  DH607:  .ASCIZ  /BAD HEADER IS :/
 12030  061770  040505  042504  020122
 12031  061776  051511  035040     000
 12032  062003     127  020104  020043  DH701:  .ASCIZ  /WD #    GOOD    BAD     HI PHY  LO PHY  VRT AD  KIPAR6/
 12033  062010  020040  043440  047517
 12034  062016  020104  020040  041040
 12035  062024  042101  020040  020040
 12036  062032  044040  020111  044120
 12037  062040  020131  046040  020117
 12038  062046  044120  020131  053040
 12039  062054  052122  040440  020104
 12040  062062  045440  050111  051101
 12041  062070  000066
 12042  062072  047507  042117  020040  DH702:  .ASCIZ  /GOOD    BAD/
 12043  062100  020040  040502  000104
 12044  062106  040506  046111  047111  DH703:  .ASCIZ  /FAILING DATA WORD :/
 12045  062114  020107  040504  040524
 12046  062122  053440  051117  020104
 12047  062130  000072
 12048  062132  051123  004460  051123  DH704:  .ASCIZ  /SR0    SR1     SR2     SR3/
 12049  062140  004461  051123  004462
```

```
12050  062146  051123  000063
12051  062152  052113  030461  051040  DH705:  .ASCIZ  /KT11 REGS :/
12052  062160  043505  020123  000072
12053  062166  042515  020115  042522  DH706:  .ASCIZ  /MEM REGS :/
12054  062174  051507  035040     000
12055  062201     120  004503  047514  DH707:  .ASCIZ  /PC     LOERAD  HIERAD  MEMSYS/
12056  062206  051105  042101  044011
12057  062214  042511  040522  004504
12058  062222  042515  051515  051531
12059  062230     000
12060  062231     120  004503  051503  DH710:  .ASCIZ  /PC     CSR AD  CSR/
12061  062236  020122  042101  041411
12062  062244  051123     000
12063  062247     103  046131  047111  DH711:  .ASCIZ  /CYLINDERS :/
12064  062254  042504  051522  035040
12065  062262     000
12066  062263     122  042113  020123  DH204:  .ASCIZ  /RKDS    RKER    RKMR2   RKMR3/
12067  062270  020040  051040  042513
12068  062276  020122  020040  051040
12069  062304  046513  031122  020040
12070  062312  051040  046513  031522
12071  062320     000
12072  062321     105  051122  052040  DH502:  .ASCIZ  /ERR TRYING TO GET FAILURE INFO/
12073  062326  054522  047111  020107
12074  062334  047524  043440  052105
12075  062342  043040  044501  052514
12076  062350  042522  044440  043116
12077  062356  000117
12078  062360  023462  042116  052040  DH503:  .ASCIZ  /2'ND TIMEOUT OCCURRED GETTING DRV STATUS/
12079  062366  046511  047505  052125
12080  062374  047440  041503  051125
12081  062402  042522  020104  042507
12082  062410  052124  047111  020107
12083  062416  051104  020126  052123
12084  062424  052101  051525     000
12085  062431     043  047440  020106  DH800:  .ASCIZ  /# OF RETRIES:/
12086  062436  042522  051124  042511
12087  062444  035123     000
12088          062450                  .EVEN
12089  062450  001116  005510  005506  DT100:  .WORD   $ERRPC,DRIVE,ERRCOM,$REG0,$REG1,$REG2,$REG3
12090  062456  001162  001164  001166
12091  062464  001170
12092  062466  001256  001260          DT102:  .WORD   $REG36,$REG37
12093  062472  001174  001176  001200  DT201:  .WORD   $REG5,$REG6,$REG7,$REG10,$REG11,$REG12,$REG13
12094  062500  001202  001204  001206
12095  062506  001210
12096  062510  001212  001214          DT202:  .WORD   $REG14,$REG15
12097  062514  001216  001220  001222  DT203:  .WORD   $REG16,$REG17,$REG20,$REG21,$REG22,$REG23,$REG24,$REG25
12098  062522  001224  001226  001230
12099  062530  001232  001234
12100  062534  001174  001176  001200  DT601:  .WORD   $REG5,$REG6,$REG7
12101  062542  001202  001204  001206  DT602:  .WORD   $REG10,$REG11,$REG12,$REG13,$REG14,$REG15,$REG16
12102  062550  001210  001212  001214
12103  062556  001216
12104  062560  005264  005262          DT103:  .WORD   PRMPH0,PRMPL0
12105
```

```
12106  062564  000006              DF01:   .WORD   6               ;NUMBER OF HEADER LINES
12107  062566     000              .BYTE   0               ;NUMBER OF COL FOR FIRST HDR
12108  062567     000              .BYTE   0               ;ALL CHARACTERS OCTAL
12109  062570  061122              .WORD   DH101           ;SECOND HDR LINE
12110  062572     007      000     .BYTE   7,0             ;NUM OF COL-ALL OCTAL
12111  062574  061211              .WORD   DH102
12112  062576     002      000     .BYTE   2,0
12113  062600  061002              .WORD   DH200
12114  062602     000      000     .BYTE   0,0
12115  062604  061314              .WORD   DH201
12116  062606     007      000     .BYTE   7,0
12117  062610  061017              .WORD   DH500
12118  062612     000      000     .BYTE   0,0
12119
12120  062614  000007              DF02:   .WORD   7
12121  062616     000      000     .BYTE   0,0
12122  062620  061122              .WORD   DH101
12123  062622     007      000     .BYTE   7,0
12124  062624  061211              .WORD   DH102
12125  062626     002      000     .BYTE   2,0
12126  062630  061002              .WORD   DH200
12127  062632     000      000     .BYTE   0,0
12128  062634  061314              .WORD   DH201
12129  062636     007      000     .BYTE   7,0
12130  062640  061403              .WORD   DH202
12131  062642     002      000     .BYTE   2,0
12132  062644  061420              .WORD   DH203
12133  062646     010      000     .BYTE   10,0
12134
12135  062650  000010              DF03:   .WORD   10
12136  062652     000      000     .BYTE   0,0
12137  062654  061122              .WORD   DH101
12138  062656     007      000     .BYTE   7,0
12139  062660  061211              .WORD   DH102
12140  062662     002      000     .BYTE   2,0
12141  062664  061002              .WORD   DH200
12142  062666     000      000     .BYTE   0,0
12143  062670  061314              .WORD   DH201
12144  062672     007      000     .BYTE   7,0
12145  062674  061050              .WORD   DH501           ;''THE FOLLOWING REG DATA MB INCORRECT''
12146  062676     000      000     .BYTE   0,0
12147  062700  061403              .WORD   DH202
12148  062702     002      000     .BYTE   2,0
12149  062704  061420              .WORD   DH203
12150  062706     010      000     .BYTE   10,0
12151
12152  062710  000003              DF04:   .WORD   3
12153  062712     000      000     .BYTE   0,0
12154  062714  061122              .WORD   DH101
12155  062716     007      000     .BYTE   7,0
12156  062720  061211              .WORD   DH102
12157  062722     002      000     .BYTE   2,0
12158
12159  062724  000007              DF05:   .WORD   7               ;OPI, HVRC
12160  062726     000      000     .BYTE   0,0
12161  062730  061122              .WORD   DH101
```

```
12162  062732  007     000          .BYTE   7,0
12163  062734  061211               .WORD   DH102
12164  062736  002     000          .BYTE   2,0
12165  062740  061514               .WORD   DH601
12166  062742  000     000          .BYTE   0,0
12167  062744  061735               .WORD   DH606
12168  062746  003     000          .BYTE   3,0
12169  062750  061533               .WORD   DH602
12170  062752  000     000          .BYTE   0,0
12171  062754  061735               .WORD   DH606
12172  062756  003     000          .BYTE   3,0
12173
12174  062760  000007          DF07:  .WORD   7
12175  062762  000     000          .BYTE   0,0
12176  062764  061122               .WORD   DH101
12177  062766  007     000          .BYTE   7,0
12178  062770  061211               .WORD   DH102
12179  062772  002     000          .BYTE   2,0
12180  062774  061570               .WORD   DH604
12181  062776  000     000          .BYTE   0,0
12182  063000  061616               .WORD   DH6041
12183  063002  003     000          .BYTE   3,0
12184  063004  061663               .WORD   DH605
12185  063006  000     000          .BYTE   0,0
12186  063010  061700               .WORD   DH6051
12187  063012  003     000          .BYTE   3,0
12188
12189  063014  000005          DF10:  .WORD   5
12190  063016  000     000          .BYTE   0,0
12191  063020  061122               .WORD   DH101
12192  063022  007     000          .BYTE   7,0
12193  063024  061211               .WORD   DH102
12194  063026  002     000          .BYTE   2,0
12195  063030  061570               .WORD   DH604
12196  063032  000     000          .BYTE   0,0
12197  063034  061616               .WORD   DH6041
12198  063036  003     000          .BYTE   3,0
12199
12200  063040  000004          DF11:  .WORD   4
12201  063042  000     000          .BYTE   0,0
12202  063044  061570               .WORD   DH604
12203  063046  000     000          .BYTE   0,0
12204  063050  061616               .WORD   DH6041
12205  063052  003     000          .BYTE   3,0
12206  063054  062003               .WORD   DH701
12207  063056  000     000          .BYTE   0,0
12208
12209  063060  000006          DF12:  .WORD   6
12210  063062  000     000          .BYTE   0,0
12211  063064  061122               .WORD   DH101
12212  063066  007     000          .BYTE   7,0
12213  063070  061211               .WORD   DH102
12214  063072  002     000          .BYTE   2,0
12215  063074  061002               .WORD   DH200
12216  063076  000     000          .BYTE   0,0
12217  063100  061314               .WORD   DH201
```

```
12218  063102  007     000          .BYTE   7,0
12219  063104  062263               .WORD   DH204
12220  063106  004     000          .BYTE   4,0
12221
12222  063110  000006          DF13:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12223  063112  000     000          .BYTE   0,0              ;IN HEADER COMPARE ERROR
12224  063114  061122               .WORD   DH101            ;AND 2ND LEVEL HEADER
12225  063116  007     000          .BYTE   7,0              ;VRC ERROR
12226  063120  061211               .WORD   DH102
12227  063122  002     000          .BYTE   2,0
12228  063124  061514               .WORD   DH601
12229  063126  000     000          .BYTE   0,0
12230  063130  061735               .WORD   DH606
12231  063132  003     000          .BYTE   3,0
12232  063134  062321               .WORD   DH502
12233  063136  000     000          .BYTE   0,0
12234
12235  063140  000006          DF14:  .WORD   6              ;FORMAT FOR 2ND LEVEL ERROR
12236  063142  000     000          .BYTE   0,0              ;IN OPERATION INCOMPLETE ERROR
12237  063144  061122               .WORD   DH101
12238  063146  007     000          .BYTE   7,0
12239  063150  061211               .WORD   DH102
12240  063152  002     000          .BYTE   2,0
12241  063154  061514               .WORD   DH601
12242  063156  000     000          .BYTE   0,0
12243  063160  061735               .WORD   DH606
12244  063162  003     000          .BYTE   3,0
12245  063164  062321               .WORD   DH502
12246  063166  000     000          .BYTE   0,0
12247
12248  063170  000011          DF15:  .WORD   11
12249  063172  000     000          .BYTE   0,0
12250  063174  061122               .WORD   DH101
12251  063176  007     000          .BYTE   7,0
12252  063200  061211               .WORD   DH102
12253  063202  002     000          .BYTE   2,0
12254  063204  061002               .WORD   DH200
12255  063206  000     000          .BYTE   0,0
12256  063210  061314               .WORD   DH201
12257  063212  007     000          .BYTE   7,0
12258  063214  061403               .WORD   DH202
12259  063216  002     000          .BYTE   2,0
12260  063220  062360               .WORD   DH503
12261  063222  000     000          .BYTE   0,0
12262  063224  061050               .WORD   DH501
12263  063226  000     000          .BYTE   0,0
12264  063230  061420               .WORD   DH203
12265  063232  010     000          .BYTE   10,0
12266
12267  063234  000011          DF16:  .WORD   11
12268  063236  000     000          .BYTE   0,0
12269  063240  061122               .WORD   DH101
12270  063242  007     000          .BYTE   7,0
12271  063244  061211               .WORD   DH102
12272  063246  002     000          .BYTE   2,0
12273  063250  061002               .WORD   DH200
```

```
12274  063252  000      000          .BYTE   0,0
12275  063254  061314                .WORD   DH201
12276  063256  007      000          .BYTE   7,0
12277  063260  061403                .WORD   DH202
12278  063262  002      000          .BYTE   2,0
12279  063264  062321                .WORD   DH502
12280  063266  000      000          .BYTE   0,0
12281  063270  061050                .WORD   DH501
12282  063272  000      000          .BYTE   0,0
12283  063274  061420                .WORD   DH203
12284  063276  010      000          .BYTE   10,0
12285
12286  063300  000005          DF17:  .WORD   5
12287  063302  000      000          .BYTE   0,0
12288  063304  061122                .WORD   DH101
12289  063306  007      000          .BYTE   7,0
12290  063310  061211                .WORD   DH102
12291  063312  002      000          .BYTE   2,0
12292  063314  062247                .WORD   DH711
12293  063316  000      000          .BYTE   0,0
12294  063320  062072                .WORD   DH702
12295  063322  002      000          .BYTE   2,0
12296
12297  063324  000002          DF20:  .WORD   2
12298  063326  000      000          .BYTE   0,0
12299  063330  061252                .WORD   DH104
12300  063332  002      000          .BYTE   2,0
12301
12302  063334  000002          DF21:  .WORD   2
12303  063336  000      000          .BYTE   0,0
12304  063340  061700                .WORD   DH6051
12305  063342  003      000          .BYTE   3,0
12306
12307  063344  000006          DF22:  .WORD   6
12308  063346  000      000          .BYTE   0,0
12309  063350  060751                .WORD   DH100
12310  063352  000      000          .BYTE   0,0
12311  063354  061122                .WORD   DH101
12312  063356  007      000          .BYTE   7,0
12313  063360  061211                .WORD   DH102
12314  063362  002      000          .BYTE   2,0
12315  063364  061272                .WORD   DH106
12316  063366  000      000          .BYTE   0,0
12317  063370  061252                .WORD   DH104
12318  063372  002      000          .BYTE   2,0
12319
12320  063374  000002          DF23:  .WORD   2
12321  063376  000      000          .BYTE   0,0
12322  063400  062431                .WORD   DH800
12323  063402  001      000          .BYTE   1,0
12324
12325  063404  000001          DF24:  .WORD   1
12326  063406  002      000          .BYTE   2,0
12327
12328  063410  000000          DF25:  .WORD   0
12329  063412  007      000          .BYTE   7,0
```

```
12330
12331  063414  000002                  DF26:   .WORD   2
12332  063416     002      000                 .BYTE   2,0
12333  063420  061420                          .WORD   DH203
12334  063422     010      000                 .BYTE   10,0
12335
12336  063424  000005                  DF27:   .WORD   5
12337  063426     000      000                 .BYTE   0,0
12338  063430  061122                          .WORD   DH101
12339  063432     007      000                 .BYTE   7,0
12340  063434  061211                          .WORD   DH102
12341  063436     002      000                 .BYTE   2,0
12342  063440  062106                          .WORD   DH703
12343  063442     000      000                 .BYTE   0,0
12344  063444  062072                          .WORD   DH702
12345  063446     002      000                 .BYTE   2,0
12346
12347  063450  000005                  DF30:   .WORD   5
12348  063452     000      000                 .BYTE   0,0
12349  063454  061122                          .WORD   DH101
12350  063456     007      000                 .BYTE   7,0
12351  063460  061211                          .WORD   DH102
12352  063462     002      000                 .BYTE   2,0
12353  063464  062152                          .WORD   DH705
12354  063466     000      000                 .BYTE   0,0
12355  063470  062132                          .WORD   DH704
12356  063472     004      000                 .BYTE   4,0
12357
12358  063474  000005                  DF31:   .WORD   5
12359  063476     000      000                 .BYTE   0,0
12360  063500  061122                          .WORD   DH101
12361  063502     007      000                 .BYTE   7,0
12362  063504  061211                          .WORD   DH102
12363  063506     002      000                 .BYTE   2,0
12364  063510  062166                          .WORD   DH706
12365  063512     000      000                 .BYTE   0,0
12366  063514  062231                          .WORD   DH710
12367  063516     003      000                 .BYTE   3,0
12368
12369  063520                          RWBUF:          ;READ/WRITE DATA BUF <AT LEAST 1536(DEC) WORDS>
12370
12371  063520  005015  020040  020040  NOTMSG: .ASCII  <15><12>/          *** NOTE ***/<15><12><12>
12372  063526  020040  020040  020040
12373  063534  025052  020052  047516
12374  063542  042524  025040  025052
12375  063550  005015     012
12376  063553     101  046114  042040          .ASCII  /ALL DRIVES TO BE TESTED MUST HAVE :/<15><12><12>
12377  063560  044522  042526  020123
12378  063566  047524  041040  020105
12379  063574  042524  052123  042105
12380  063602  046440  051525  020124
12381  063610  040510  042526  035040
12382  063616  005015     012
12383  063621     061  020056  020101          .ASCII  /1. A 20 OR 22 SECTOR FORMATTED CARTRIDGE/<15><12>
12384  063626  030062  047440  020122
12385  063634  031062  051440  041505
```

```
12386  063642  047524  020122  047506
12387  063650  046522  052101  042524
12388  063656  020104  040503  052122
12389  063664  044522  043504  006505
12390  063672     012
12391  063673     062  020056  042510          .ASCII  /2. HEADS MANUALLY LOADED/<15><12>
12392  063700  042101  020123  040515
12393  063706  052516  046101  054514
12394  063714  046040  040517  042504
12395  063722  006504     012
12396  063725     063  020056  042504          .ASCII  /3. DESIRED PORT SELECTED/<15><12>
12397  063732  044523  042522  020104
12398  063740  047520  052122  051440
12399  063746  046105  041505  042524
12400  063754  006504     012
12401  063757     064  020056  051127          .ASCII  /4. WRITE LOCK DISABLED/<15><12>
12402  063764  052111  020105  047514
12403  063772  045503  042040  051511
12404  064000  041101  042514  006504
12405  064006     012
12406  064007     065  020056  051104          .ASCII  /5. DRIVE READY LIGHT ON:/<15><12><12>
12407  064014  053111  020105  042522
12408  064022  042101  020131  044514
12409  064030  044107  020124  047117
12410  064036  005015     012
12411  064041     104  044522  042526          .ASCII  /DRIVES NOT TO BE TESTED MUST HAVE BOTH/<15><12>
12412  064046  020123  047516  020124
12413  064054  047524  041040  020105
12414  064062  042524  052123  042105
12415  064070  046440  051525  020124
12416  064076  040510  042526  041040
12417  064104  052117  006510     012
12418  064111     120  051117  051524          .ASCII  /PORTS DE-SELECTED./<15><12><12>
12419  064116  042040  026505  042523
12420  064124  042514  052103  042105
12421  064132  006456  005012
12422  064136  046120  040505  042523          .ASCII  &PLEASE NOTE THAT THE DEFAULT VERSION OF READ/WRITE&<15><12>
12423  064144  047040  052117  020105
12424  064152  044124  052101  052040
12425  064160  042510  042040  043105
12426  064166  052501  052114  053040
12427  064174  051105  044523  047117
12428  064202  047440  020106  042522
12429  064210  042101  053457  044522
12430  064216  042524  005015
12431  064222  040504  040524  052040          .ASCII  /DATA TEST 21 TAKES ABOUT 10 MIN. (20 FOR RK07)./<15><12>
12432  064230  051505  020124  030462
12433  064236  052040  045501  051505
12434  064244  040440  047502  052125
12435  064252  030440  020060  044515
12436  064260  026116  024040  030062
12437  064266  043040  051117  051040
12438  064274  030113  024467  006454
12439  064302     012
12440  064303     122  043505  051101          .ASCII  /REGARDLESS OF THE PASS NUMBER./<15><12><12>
12441  064310  046104  051505  020123
```

```
 12442  064316  043117  052040  042510
 12443  064324  050040  051501  020123
 12444  064332  052516  041115  051105
 12445  064340  006456  005012
 12446  064344  047506  020122  030065          .ASCIZ  /FOR 50 HZ CPU OPERATION, PATCH LOC. 166 = 1./<15><12><12>
 12447  064352  044040  020132  050103
 12448  064360  020125  050117  051105
 12449  064366  052101  047511  026116
 12450  064374  050040  052101  044103
 12451  064402  046040  041517  020056
 12452  064410  033061  020066  020075
 12453  064416  027061  005015  000012
 12454
 12455
 12456
 12457          000001                  .END
```

```
ABASE = 177440          2026#   2118    2159    4562
ABORT = 000200          4085#   6970    8883    9240    9245    9328    9353    9441    9451    9456    9461    9466    9474
                        9480    9490    9528
ABOVE   012005          4362#   5843
ABOVE1  012042          4367#   5936
ABOVE3  012142          4379#   6090
ABVMX1  003172          3472#   5816*   5841    5906*   6061*   7729*   7910
ABVMX2  003174          3473#   5926*   6079*   7730*   7925
ACDW1 = 000000          2118
ACDW2 = 000000          2118
ACLO  = 000010          3073#   9290
ACPUOP= 000000          2118    2133
ADDW0 = 000000          2118
ADDW1 = 000000          2118
ADDW10= 000000          2118
ADDW11= 000000          2118
ADDW12= 000000          2118
ADDW13= 000000          2118
ADDW14= 000000          2118
ADDW15= 000000          2118
ADDW2 = 000000          2118
ADDW3 = 000000          2118
ADDW4 = 000000          2118
ADDW5 = 000000          2118
ADDW6 = 000000          2118
ADDW7 = 000000          2118
ADDW8 = 000000          2118
ADDW9 = 000000          2118
ADEVCT= 000000          2118    2124
ADEVM = 000377          2027#   2118    2160
AENV  = 000000          2118    2129
AENVM = 000000          2118    2130
AFATAL= 000000          2118    2121
ALDRVS  010124          4178#   4629
ALLDRV  014372          4568    4638    4739#
ALLTRM  043724          6971    8891    9329    9464    9471#
ALNCYL= 000365          4078#
ALNPAK  007663          4149#   7036
AMADR1= 000000          2118    2146
AMADR2= 000000          2118    2150
AMADR3= 000000          2118    2153
AMADR4= 000000          2118    2156
AMAMS1= 000000          2118    2140
AMAMS2= 000000          2118    2148
AMAMS3= 000000          2118    2151
AMAMS4= 000000          2118    2154
AMSGAD= 000000          2118    2126
AMSGLG= 000000          2118    2127
AMSGTY= 000000          2118    2120
AMTYP1= 000000          2118    2141
AMTYP2= 000000          2118    2149
AMTYP3= 000000          2118    2152
AMTYP4= 000000          2118    2155
ANYDER= 100000          4091#   7028    8951    9516
APASS = 000000          2118    2123
APRIOR= 000000          2118
```

```
APTCSU= 000040          10853   11513#
APTENV= 000001           6427   10846   11292   11469   11511#
APTPAS  025516           6422    6427#
APTSIZ= 000200           4485   11510#
APTSPO= 000100          10848   11471   11512#
ASKMDE  015272           4907#   4912    4913    4918    4933    4938    4943    4960    4982    5006    5020    5031    5040
                         5056    5075    5084    5110    5133    5205    7359
ASKTMD  014526           4789#   4792    4793    4798    4822    4830    4836    4852    4863    4872    4897
ASKTST  014522           4705    4788#
ASWREG= 000000           2118    2131
AS2SP2  056320           8998   11681#
ATESTN= 000000           2118    2122
AUNIT = 000000           2118    2125
AUSWR = 000000           2118    2132
AVECT1= 120210           2025#   2118    2157    4561
AVECT2= 000000           2118    2158
AVGEQ   011735           4354#   7995
AVGSEK  011560           4331#   5985
A.ABNL  003046           3354#   6889*   6977*   6981*   8446*   8535*   8552*   8561*   9216*   9472*   9485*   9487*   10414
A.CONT  003050           3355#   10424
A.NORM  003044           3353#   6890*   9217*   9473*   10418
BADDRV  007611           4141#   6984*   6985*   6986    6998*   6999*   7000    7006*   7007*   7008    7033*   7034*   7035
BADSEC= 001000           4087#   8253    8824    8832    9418    9521
BAD632  012307           4398#   7030
BAI   = 000020           3032#   10583
BA16  = 000400           3008#
BA17  = 001000           3009#
BDSCHD  037074           8535    8552    8566#
BDSECT  011411           4310#   8584
BDSRCK  040146           8251    8814#   9417
BELOW   011750           4357#   5837
BGNRTY  044116           9140    9483    9489    9514#
BIT0  = 000001           1917#   3005    3022    3048    3070    3218    3362    3455    4646    4744    6158    7398    8127
                         8144    8159    8198    8215    8269    8311    8868    8873    9095   10698
BIT00 = 000001           1907#   1917
BIT01 = 000002           1906#   1916    5717
BIT02 = 000004           1905#   1915
BIT03 = 000010           1904#   1914    5830    5933    6087
BIT04 = 000020           1903#   1913
BIT05 = 000040           1902#   1912
BIT06 = 000100           1901#   1911
BIT07 = 000200           1900#   1910    5462    7526    7713
BIT08 = 000400           1899#   1909    7554
BIT09 = 001000           1898#   1908    5703   11302   11433
BIT1  = 000002           1916#   3023    3050    3219    3363    3456    4079    6326    9099
BIT10 = 002000           1897#   3010    3039    3061    3096    3110    3123    3138    3152    3231    3372    4088    9130
                        11280
BIT11 = 004000           1896#   3011    3040    3062    3081    3097    3111    3124    3139    3153    3232    4089   11440
BIT12 = 010000           1895#   3013    3041    3063    3098    3112    3125    3126    3140    3154    3233    4090    8871
                         9010    9038
BIT13 = 020000           1894#   3014    3042    3064    3082    3099    3113    3127    3141    3155    3234    6869    6871
                         8150    8153    8224    8227   11287
BIT14 = 040000           1893#   3015    3043    3065    3083    3100    3114    3128    3142    3156    3235    3373   11415
BIT15 = 100000           1892#   3016    3017    3044    3066    3084    3101    3161    3236    3374    4091    5017    7289
                         9134
BIT2  = 000004           1915#   3024    3051    3052    3072    3220    3364    3457    4080    6332    9103
```

```
BIT3   = 000010  1914# 3030  3031  3053  3073  3221  3365  3458  4081  6328  9110
BIT4   = 000020  1913# 3025  3032  3054  3074  3075  3090  3132  3146  3222  3366  3459  4082
                 5315  6310  9114
BIT5   = 000040  1912# 3033  3034  3055  3076  3091  3105  3118  3133  3147  3223  3367  4083
                 6269  6316  6469  8975  9118  10495
BIT6   = 000100  1911# 3006  3035  3056  3077  3092  3106  3119  3134  3148  3224  3369  4084
                 4829  5019  5039  6166  6398  6474  6507  8296
BIT7   = 000200  1910# 3007  3036  3057  3078  3079  3093  3107  3120  3135  3149  3226  4085
                 4559  4741  5291  5294  6988  8869  8896  9618  9621
BIT8   = 000400  1909# 3008  3037  3058  3059  3080  3094  3108  3121  3136  3150  3227  3370
                 4086  9122
BIT9   = 001000  1908# 3009  3038  3060  3095  3109  3122  3137  3151  3228  3371  4087  5300
                 6653  9126  11392
BKSLSH  012410   4411# 7155  7163
BLDEXH  044756   9369  9394  9665#
BLWMIN  003170   3471# 5810* 5835  7728*
BPTVEC= 000014   1924#
BSE    = 000200  3057# 8566  9409  9848  10014 10250
BSERR  = 000002  4079# 6148  8746  9411  9421  9514
BSFACT  004232   3489# 8534  8582  8815
BSSOFT  003232   3488# 8546  8615  8820  8822
BUFFO   005272   3498# 4605  4636  4639  4688  4709  4713  4715  4720  4731  4735  4796  4799
                 4802  4805  4840  4842  4873  4876  4878* 4883  4901  4916  4919  4922  4925
                 4928  4974  4977  4980  5060  5062  5094  5111  5137  5144  5154  5156  5159
                 5164  5173  5211  5214  6598  6602  7158  7172* 7177* 7178  7182* 7299  7301
                 7321  7324  7326* 7331  7347
BUFPRT  005502   3501#
B.BA16= 000001   3022# 10577
B.BA17= 000002   3023# 10577
B.CDT  = 000004  3024#  5263  6974  10231 10577
B.CFMT= 000020   3025#  5302  5748  5756  5776  7025  7493  8524  9623  10231 10577 10661
CALBRT  032646   5741  5882  5964  6036  7640#
CART    011342   4302# 7503
CCLR   = 100000  3017# 10184
CDT    = 002000  3010# 6959  9990  10540 10597 10648 10671
CERR   = 100000  3016# 9378  9403  9886  9945  10004 10186 10237 10552 10608 10696
CFMT   = 010000  3013# 5757  5777  9644  9677  9990  10540 10597 10648 10671
CHGTST  014744   4804  4833# 4837  4844
CHKITR  030464   5349  5373  5399  5447  5486  5534  5590  5626  5650  5683  5733  5874  5956
                 6028  6105  6188  6263  7071#
CHKLIM  035716   6281  6308  8330#
CHKLST  014450   4612  4694  4737  4754#
CHKPRM  031564   5100  5179  7380#
CHKXDP  026726   4962  6744#
CLEAR  = 000105  2976#
CLKFAL  011517   4325# 7652
CLKOF   032624   7617# 7694
CLKON   032550   7596# 7647
CLOCK   032620   4532  7611#
CLR    = 000040  3033#
CLRPRM  027556   6894  6913#
CMDTO  = 000100  3224# 9263  9578  9747
CMPBUF  035010   6334  8185#
CMPTIM  033560   5899  5919  6054  6072  7848#
CMSTRT  012534   4436  4440  4445#  9505
CNTRDY  010032   4168# 4690  8459
```

```
CNTRLC  012353           4405#   6473    7126
CNTRLG  012377           4409#   6487    7144
CNTRLR  012365           4407#   6483
CNTRLU  012372           4408#   7138
CNTRLZ  012360           4406#   6479    7132
CNVTIM  034072           5783    7771    7948#
COE   = 001000           3060#   9335    9848   10014   10250
COMMA   012412           4412#   4651
COMSTR  005246           3492#   8916    8931    9533
CONCLR= 000176           2994#   8885   10679
CONERR  041414           3355    9091#
CR    = 000015           1832#   4095    4098    4376    4385   10892   10902
CRLF  = 000200           1833#  10863   10902
CRMPHO  005270           3497#   8259    8926    8928*
CRMPLO  005266           3496#   8260    8927    8929*
CRTSER  032210           5314    7492#
CR2LF   012404           4410#   8997    9009    9033
CS      005770           3621#   5315    5830    5933    6087    8975
C1LOUT  036266           5788    5896    5916    6051    6069    8430#   8860
CTC   = 004000           3011#   9298   10246
CYLNDR  005514           3506#   5459*   5462    5465    5467*   5632    5688*   5695    5702*   5703    5706*   5713    5717
                         7582*   7583*   7584    7586*
C.INIT  050534           7767    8901   10457#
C.RTRN  052406          10561   10589   10617   10624   10627   10634   10638   10655   10677   10682   10692   10701   10714
                        10718   10723#
C.SPEC  051426          10497   10593#
DASTRT  012504           1989    4438#
DCK   = 100000           3066#   8566    9423    9848   10014   10250
DCKERR= 000020           4082#   9425    9514
DCLO  = 000020           3075#
DDISP = 177570           1839#   2056    4473
DDT   = 000400           3080#
DERCNT  003121           3431#   9362    9387    9432    9443
DESL  = 000010           3030#
DFLTST  005670           3581#   3601    4827    4846    4849    4870    4895
DFQUES  010775           4260#   5053
DFSTRT  012450           1983    4430#
DFTEST  010335           4205#   4833
DF01    062564           2183    2189    2195    2201    2243    2351    2621    2657   12106#
DF02    062614           2207    2225    2231    2237    2249    2255    2261    2267    2273    2279    2285    2291    2297
                         2303    2309    2315    2369    2375    2381    2489    2495    2501    2507    2513    2519    2525
                         2531    2537    2543    2585    2603    2627   12120#
DF03    062650           2213    2219    2591    2597   12135#
DF04    062710           2345   12152#
DF05    062724           2321    2327   12159#
DF07    062760           2333   12174#
DF10    063014           2339   12189#
DF11    063040           2639   12200#
DF12    063060           2387    2393    2399    2405    2411    2417    2423    2429   12209#
DF13    063110           2435    2447   12222#
DF14    063140           2441   12235#
DF15    063170           2453   12248#
DF16    063234           2459   12267#
DF17    063300           2477   12286#
DF20    063324           2645   12297#
DF21    063334           2555   12302#
```

```
DF22    063344          2651  12307#
DF23    063374          2561   2567    2573  12320#
DF24    063404          2579  12325#
DF25    063410          2483   6155*   8187*  8241*  12328#
DF26    063414          2633  12331#
DF27    063424          2609  12336#
DF30    063450          2663   6671*  12347#
DF31    063474          2669  12358#
DH100   060751          2181   2187    2193   2199   2205   2211   2217   2223   2229   2235   2241   2247   2253
                        2259   2265    2271   2277   2283   2289   2295   2301   2307   2313   2319   2325   2331
                        2337   2343    2349   2367   2373   2379   2385   2391   2397   2403   2409   2415   2421
                        2427   2433    2439   2445   2451   2457   2475   2487   2493   2499   2505   2511   2517
                        2523   2529    2535   2541   2583   2589   2595   2601   2607   2619   2625   2655   2661
                        2667  11932#  12309
DH101   061122          9014  11952#  12109  12122  12137  12154  12161  12176  12191  12211  12224  12237  12250
                       12269  12288   12311  12338  12349  12360
DH102   061211          9025  11962#  12111  12124  12139  12156  12163  12178  12193  12213  12226  12239  12252
                       12271  12290   12313  12340  12351  12362
DH103   061227          2643  11965#
DH104   061252         11969# 12299   12317
DH105   060767          9012  11935#
DH106   061272         11972# 12315
DH200   061002         11937# 12113   12126  12141  12215  12254  12273
DH201   061314         11975# 12115   12128  12143  12217  12256  12275
DH202   061403         11985# 12130   12147  12258  12277
DH203   061420         11988# 12132   12149  12264  12283  12333
DH204   062263         12066# 12219
DH500   061017         11940# 12117
DH501   061050         11945# 12145   12262  12281
DH502   062321         12072# 12232   12245  12279
DH503   062360         12078# 12260
DH601   061514         11998# 12165   12228  12241
DH602   061533         12001# 12169
DH604   061570         12006# 12180   12195  12202
DH6041  061616          8585  12010#  12182  12197  12204
DH6042  061645          2577  12014#
DH605   061663          2553  12017#  12184
DH6051  061700         12020# 12186   12304
DH606   061735         12025# 12167   12171  12230  12243
DH607   061763         12029#
DH701   062003          6154*  8186*   8240*  12032# 12206
DH702   062072          2631  12042#  12294  12344
DH703   062106         12044# 12342
DH704   062132          6672* 12048#  12355
DH705   062152         12051# 12353
DH706   062166         12053# 12364
DH707   062201         12055#
DH710   062231         12060# 12366
DH711   062247         12063# 12292
DH800   062431         12085# 12322
DI    = 040000          3015#  7796   10034  10706
DISPLA  001142          2056#  4473*   4481* 11279* 11453*
DISPRE  000174          1980#  4481
DLT   = 100000          3044#  9355    9842   9972  10008  10248
DLTFLG  003142          3448#
DMD   = 000040          3091#
```

```
DONE    003123          3433#  8861*  8904   8950*  9092*  9210*  9547*  9550*
DOTIM   003133          3441#  4527*  4534*  5738   5879   5961   6033   7654*
DRA   = 000001          3070#  9837   10620
DRDY  = 000200          3079#
DRIV    011335          4301#  7461
DRIVE   005510          3504#  4757*  5222*  5229*  5230   5233   5308   6447   6897   6898   6944   6954   6984
                        6998   7006   7033   8867   8876   8886   8894   12089
DRNAFG  003137          3445#  8862*  9105
DRNAVL= 010000          4090#
DROPDR  011251          4291#  8981
DROT  = 000040          3076#  9345
DRPAR = 000010          3053#  9285
DRPDRV= 002000          3231#  10482
DRVCAL  040252          5289   5356   5750   5888   5970   6042   6128   6140   6143   6203   6219   6338   6896
                        6979   7013   7015   7027   7055   7460   7502   7529   7538   7780   8033   8449   8488
                        8491   8537   8554   8742   8857#  9486   9532   9542   9577   9614   9616
DRVDSC= 000040          3223#  9453   9922   10088  10150
DRVERS  003130          3438#  4493*  5227*  8974*  8978   8980*  9494*
DRVHRD= 000020          3222#  9448   9929   10124  10255
DRVLST  005606          3534#  4647   4666   4678   4725   4733*  4745*  4748*  4755   4767*  5234
DRVMSK= 000007          3029#  9831   9872   10023  10468
DRVNO   011331          4300#  5308*  5309*
DRVPDT= 000004          3220#  10030  10098
DRVPOS= 000002          3219#  10030  10057  10519
DRVSEQ  007575          4138#  4643
DRVSER  032070          5313   7457#
DRVSZD= 010000          3233#  9276   9860   10622
DRVTST  013660          4433   4610   4626#  4631   4632   4633   4641   4698   4699   4791   4819   4835   4862
                        4911   5003   5028   5055   5083   5132   5204   7356   8438
DRVUSE= 000001          3218#  10482
DRY   = 000200          3078#
DR.CLR= 000005          3001#  9910   10073  10115  10139  10253
DR.SEL= 000001          3000#  9980   10059  10374  10381  10388  10395  10599
DROXDP  010075          4174#  6951
DSC   = 040000          3083#
DSWR  = 177570          1838#  2055   4472
DTBAII= 100000          3236#  10482  10581
DTE   = 010000          3063#  8566   9350   10250
DTYE  = 000040          3055#  6972   9313   10250
DTYLST  005616          3545#  4762*  5238
DT100   062450          2182   2188   2194   2200   2206   2212   2218   2224   2230   2236   2242   2248   2254
                        2260   2266   2272   2278   2284   2290   2296   2302   2308   2314   2320   2326   2332
                        2338   2344   2350   2368   2374   2380   2386   2392   2398   2404   2410   2416   2422
                        2428   2434   2440   2446   2452   2458   2476   2488   2494   2500   2506   2512   2518
                        2524   2530   2536   2542   2584   2590   2596   2602   2608   2620   2626   2650   2656
                        2662   2668   12089#
DT102   062466          12092#
DT103   062560          2644   12104#
DT201   062472          12093#
DT202   062510          2632   12096#
DT203   062514          12097#
DT601   062534          2554   2560   2566   2572   2578   2638   12100#
DT602   062542          2482   12101#
DUACES  010615          4238#  4504
DULACS  003136          3444#  4431*  4438*  4443*  4502   4703   5319   6335   6746   7073   8864
DUNPAS  025370          5232   6397#
```

```
DZR6M   007226            4093#   4495
ECCNC = 000040            4083#   9428
ECCW  = 020000            3099#
ECH   = 000100            3056#   9426    9848   10014   10250
ECOBAD  026006            4823    5007    5032    6521#
EMTVEC= 000030            1927#   4456*   4457*
EM1     056325            2180   11682#
EM10    056474            2222   11703#
EM100   060230            2570   11868#
EM101   060253            2576   11872#
EM102   060331            2582   11880#
EM103   060375            2588   11887#
EM104   060423            2594   11891#
EM105   060445            2600   11895#
EM106   060476            2606   11900#
EM107   060531            2612   11905#
EM11    056503            2228   11705#
EM110   060574            2618   11911#
EM111   060627            2624   11916#
EM112   060650            2630   11919#
EM113   060672            2654   11922#
EM114   060720            2660   11926#
EM115   060735            2666   11929#
EM12    056516            2234   11707#
EM13    056530            2240   11709#
EM14    056540            2246   11711#
EM15    056560            2252   11714#
EM16    056574            2258   11716#
EM17    056604            2264   11718#
EM2     056344            2186   11685#
EM20    056621            2270   11721#
EM21    056634            2276   11723#
EM22    056645            2282   11725#
EM23    056660            2288   11727#
EM24    056675            2294   11730#
EM25    056711            2300   11733#
EM26    056730            2306   11736#
EM27    056742            2312   11738#
EM3     056362            2192    2672   11688#
EM30    056757            2318    2438   11741#
EM31    056771            2324    2444   11743#
EM32    057005            2330   11746#
EM33    057022            2336   11749#
EM34    057036            2342    2642   11751#
EM35    057056            2348   11754#
EM36    057110            2354   11759#
EM37    057137            2360   11763#
EM4     056400            2198   11691#
EM40    057200            2366   11769#
EM41    057244            2372   11775#
EM42    057274            2378   11779#
EM43    057324            2384   11783#
EM5     056417            2204   11694#
EM52    057355            2426   11788#
EM53    057372            2432   11791#
EM56    057407            2450    2456   11794#
```

```
EM6     056436            2210   11697#
EM60    057426            2462   11797#
EM61    057460            2468   11802#
EM62    057510            2474   11806#
EM63    057524            2390    2486   11808#
EM64    057562            2396    2492   11813#
EM65    057610            2402    2498   11817#
EM66    057632            2408    2504   11820#
EM67    057651            2414    2510   11823#
EM7     056455            2216   11700#
EM70    057710            2420    2516   11829#
EM71    057755            2522   11836#
EM72    060010            2528   11841#
EM73    060043            2534   11846#
EM74    060063            2540   11849#
EM75    060075            2546   11851#
EM76    060125            2552   11856#
EM77    060207            2558    2564   11865#
ENTLCI  010307            4201#   4789
ENTPAS  011203            4284#   5202
EQUALS  012436            4421#   7258
ERRCNT  003126            3436#   4432*   4609*   7052*   8953*   8955    8958    8960*   9095*   9211*   9495*   9503*   9523
                          9526
ERRCOM  005506            3503#   9148*   9149*  12089
ERRFRE  040652            3353    6890    8950#   9473
ERRHDL  042130            3354    6442    6889    6981    8446    8561    8568    9209#   9472    9487
ERRLMT  003127            3437#   9523
ERRVEC= 000004            1920#   4470    4471*   4482*   4490*   4491*   6666*   6667*  11420   11421*  11423*  11426*  11534
                         11535   11539*  11544*  11556*  11561*  11570*  11580*  11581*
ER2ENT  042150            9213#   9275    9383    9408    9492
EXPLAN  010441            4218#   4906
E.CCLR= 000001            3362#  10189
E.CERR= 001000            3371#  10263
E.CLAT= 000020            3366#   9915   10079   10120   10144
E.CMTO= 040000            3373#
E.CONT  003052            3356#   9095    9099    9103    9110    9114    9118    9122    9126    9130    9134    9139*   9814*
                          9852*   9915*   9925*   9996*  10018*  10041*  10047*  10079*  10120*  10144*  10189*  10241*  10263*
                         10402*  10636*  10690*  10698*  10711*
E.DLT = 000400            3370#   9996
E.DPAR= 002000            3372#   9925   10402
E.ILLD= 000100            3369#  10690
E.MDS = 100000            3374#   9814   10241   10636
E.NOAT= 000002            3363#  10041
E.SCLR= 000040            3367#  10711
E.UATT= 000004            3364#  10047
E.UDAT= 000010            3365#   9852   10018
E.UNLD= 020000            3234#   9933   10053   10128   10138
FACTBS  011364            4306#   8583
FC      005732            3606#   5379    5492    5495    5505    5508    5512    5540    5543    5545    5555    5558    5562
                          5566    5595    5601    6111    6113    6194    6297    6365
FDTERM  033450            5978    7806#
FINADR  034404            8070#   8246
FINCYL  005574            3525#   8100*   8248    8273    8279
FINMEM  037272            8627#   8677
FINSEC  005577            3527#   8088*   8250    8277    8281
FINTRK  005576            3526#   8098*   8249    8275    8280
```

```
FMTE   = 000020          3054#   9317   *0250
FNDXDP   026654          6720#   6768
FORMAT   003125          3435#   5296*   5302*   6121    6899    7494    8080    8340    8529    8549    8654    9624
FORWRD   011663          4344#   6005    7904
FS       005516          3507#   5297*   5303*   6196    6305
FT       005740          3609#   4939    6112    6114    6304
GETAVG   033632          5931    6085    7870#
GETPRM   026206          4578    4583    4592    6584#   6624
GETXDP   027210          4547    6377    6819#   8441
GNS    = ****** U        1979   11667   11668   11669   11670   11671   11674   11675   11676   11677
GO     = 000001          3005#   6957    6959
GTPKAD   044632          9434    9445    9634#
GTPRMS   037644          8722#   8754
GTSWRG   026316          4522    6620#   7145    8464
GTO      037656          8718    8721    8725#
HCRC   = 000400          3058#
HDR.AD   003070          3396#   9966    9970*  10660*
HDR.CT   003072          3397#   9974*  10663*  10666*
HIERAD= 177742           1971#
HIGOCT   005500          3500#   5098*   5178*   7229*   7232*   7385    7400    7405    7412
HLPOVL   003146          3453#   4494*
HLTPRG   044100          4671    6674    8460    8545    9501#   9529
HLTRQD   010062          4172#   8458
HOLD1    017712          5250*   5273*   5328#   7586
HOLD2    017714          5251*   5274*   5329#   5411
HOLD3    017716          5252*   5275*   5290    5330#   9617
HOLD4    017720          5253*   5276*   5293    5331#   9620
HOLD5    017722          5254*   5277*   5332#   5759
HOLD6    017724          5255*   5278*   5333#   5991
HOLD7    017726          5256*   5279*   5334#   5992
HT     = 000011          1830#  10861   10902
HVRC   = 000400          3059#   8566    9384    9848   10014   10250
HVRCER= 000004           4080#   9386    9514
HZ       000166          1985#   7601    7683
H.HEAD   003075          3400#
IC       005736          3608#   5497    5501    5507    5511    5547    5551    5557    5561    5603    5604    5608    5609
                         6367    6371
IDAE   = 002000          3061#   9340
IE     = 000100          3006#   4691    6953    6992    7783    9374    9399   10194   10258   10546   10587   10653   10720
ILC    = 000001          3048#   9301   10250
ILF    = 000004          3051#   9309
INCRMA   036562          8500#
INCRSC   037356          8639    8653#   8681
INITMM   026350          6640#   6685    6846
INITSS   027450          4693    5287    5306    5352    5376    5402    5450    5489    5537    5593    5629    5653    5686
                         5736    5877    5959    6014    6031    6108    6191    6266   6889#    6943    6975    7458    7492
                         8523
INIVRB   033140          5744    5885    5967    6039   7728#
INPUTP   015266          4706    4807    4906#   8454
INTCHR   005532          3513#   4815    4817    4820    4999    5001    5004    5024    5026    5029   5228*   5318*   6470*
                         6489   6506*   6521    8432    8436   8443*   8452    8456    8462
INTMSK   003101          3405#   9889    9913    9934    9959   10045   10077   10118   10129   10142  10413*  10417*  10421*
                        10470*  10556
INTR   = 000300          2996#
IOTVEC= 000020           1925#   4454*   4455*
IR     = 000100          3035#
```

```
IS       005756          3616#   6356
IT       005744          3611#   6360
I.AERR   047116         10065   10071   10089   10098#
I.ATTN   046446          9825    9869   10002#
I.CCLR   047502          9903    9953   10184#  10256   10616   10681
I.CSIS   050134          9896   10331#
I.CST1   050156          6967    8888    9900    9952   10335#  10605
I.DRV    003102          3409#   8892   10470   10471
I.ERR    045670          9897    9901#
I.ERRA   045650          9896#   9964
I.ERRC   045656          9834    9887    9899#  10032
I.HDAL   046114          9880    9943#
I.INTR   045234          4508    5848    5857    7776    7795    9804#
I.ISRL   003074          3399#   9818    9821*   9824*  10195*  10492*  10646*  10719*
I.ISSU   047564          9911    9981   10060   10074   10116   10140   10226#  10375   10382   10389   10396
I.IOO    045514          9822    9839    9866#
I.RTRN   047464          9816    9854    9864    9892    9904    9909    9912    9918    9927    9936    9939    9954    9956
                         9982    9985    9994    9998   10020   10036   10043   10049   10061   10075   10082   10094   10114
                        10117   10122   10131   10134   10141   10146   10155#
I.STAT   050240          9908   10113   10372#  10626
I.STOR   050006         10236   10286#
I.UNLD   047364         10055   10138#
KBDHDL   025626          4505    6462#
KIPAR0=  172340          1958#   6642   11520   11548
KIPAR1=  172342          1959#
KIPAR2=  172344          1960#
KIPAR3=  172346          1961#
KIPAR4=  172350          1962#
KIPAR5=  172352          1963#
KIPAR6=  172354          1964#   6854*   6872*   8143*   8152*   8214*   8222    8226*
KIPAR7=  172356          1965#  11564
KIPDR0=  172300          1947#
KIPDR1=  172302          1948#
KIPDR2=  172304          1949#
KIPDR3=  172306          1950#
KIPDR4=  172310          1951#
KIPDR5=  172312          1952#
KIPDR6=  172314          1953#
KIPDR7=  172316          1954#
KTERHD   026436          4510    6662#
LASTWC   005600          3528#   8071    8245*
LC       005734          3607#   5244*   5267*   5358    5382    5495    5498    5502    5505    5545    5548    5552    5555
                         5596    5601    5605    5610    6365    6368    6372
LCM1     017704          5242*   5265*   5325#   8376
LCM5     017702          5241*   5264*   5324#   8331
LCP1     017710          5249*   5272*   5327#   5972    6352
LCVEC    003160          3466#   4539
LEV2ER=  000400          4086#   9213    9268    9365    9371    9390    9396    9488    9548
LF     = 000012          1831#   4095    4098    4376    4385   10896   10902
LIM1     012261          4393#   5838
LIM2     012274          4395#   5844
LKS      003150          3462#   4538    7598*   7619*
LODBUF   034562          6314    8118#
LODFLS   015010          4675    4841    4846#
LODFPT   016172          4853    5061    5066#
LODP14   034346          6271    8047#
```

```
LODSEC  034240            6127   6138   6202   6217   8009#  8031
LOERAD= 177740            1970#
LOWOCT  005476            3499#  5097*  5177*  7228*  7233   7390   7392   7395   7398   7403   7408   7411
LS      005520            3508#  5298*  5304*  6357
LSTCYL  017706            5245*  5268*  5326#  5907   5930   5983   6045   7498   7584   8336   8365   8388   8526
LSTMEM  007354            4110#  6567
LSTTRK= 000002            4077#  7022   7499
LSTTST  014626            4801   4810#
LT      005742            3610#  4939   6362
MA      005762            3618#  4951   4953   6285   6286   6289   6290   6753   6764   6766   6771   6779   7435
                          7437   8510*  8511*  8512*  8719   8720
MAHILM  006126            3677#  4954   4957   6539*  6540*  6554*  6555*  6568   6720   6724   6726   6791   6794
                          7433   7434
MAPH00= 170202            1968#  8928
MAPL00= 170200            1967#  6702   8929
MAWRDC  010651            4243#  7415
MAXEQ   011726            4352#  7982
MAXHI1= 000000            4064#  5811
MAXHI2  007220            4068#  5258*  5282*  5901   5921
MAXHI3= 000000            4070#
MAXHI4  007224            4074#  5260*  5284*  6056   6074
MAXIH1  003220            3483#  5797   5802*  5998   7738*  7836*
MAXIH2  003230            3487#  6000   7742*  7833*
MAXIL1  003216            3482#  5800   5803*  5839   5997   7737*  7908
MAXIL2  003226            3486#  5999   7741*  7835*  7923
MAXLO1= 062031            4063#  5814
MAXLO2  007216            4067#  5257*  5281*  5904   5924
MAXLO3= 112160            4069#
MAXLO4  007222            4073#  5259*  5283*  6059   6077
MAXSEK  011636            4340#  6089
MCLK  = 000400            3094#
MDFLAG  003116            3428#  4430*  4439*  4442*  4555   4673   4851   5071   6949   7075   8434
MDFY15  011130            4276#  7292
MDS   = 001000            3038#  6963   9812   10239  10614
MEMABT  003145            3451#
MEMCSR= 172100            1969#
MEMSYS= 177744            1972#
MEMVEC= 000114            3468#
MERD  = 001000            3095#
MESMSK= 000017            3088#
MEWD  = 002000            3096#
MICROS  011744            4356#  7973   7986   7999
MIDXFR  037422            8676#  8749
MIND  = 000200            3093#
MINEQ   011717            4350#  7969
MINHI1= 000000            4062#  5805
MINIH1  003214            3481#  5790   5795*  7736*
MINIH2  003224            3485#  7740*
MINIL1  003212            3480#  5793   5796*  5833   5898   6053   7735*  7905
MINIL2  003222            3484#  5918   6071   7739*  7920
MINLO1= 057467            4061#  5808
MMVEC = 000250            1936#  4510*  4511*
MODP15  031240            5103   5182   7281#
MSP   = 000100            3092#
MXWRDC  032014            7414   7433#
M.CADD= 017760            3164#
```

```
M.CDIF= 017760  3163#
M.DPID  053210  5993  6001  7689  7877  7884  7959  10945#
M.DPIM  053150  7812  7832  7950  8360  8372  8382  10916#
M.DP01  053160  10919#  10928
M.DP02  053200  10923  10927#
M.DP40  053246  10956#  10977
M.DP41  053302  10961  10968#
M.DP42  053312  10967  10972#
M.DP44  053344  10979  10984#
M.DP50  053356  10954  10988#
M.DRV = 000007  3160#
M.HEAD= 007000  3167#
M.ID  = 000003  3162#
M.PAR = 100000  3161#
M.SECT= 000760  3166#
M.SER = 077770  3165#
NCYL1   005522  3509#  5492*  5493  5497*  5498  5501*  5502  5507*  5508  5511*  5512  5514  5540*
                5541  5547*  5548  5551*  5552  5557*  5558  5561*  5562  5564  5595*  5597  5603*
                5605  5608*  5610  5971*  5973*  5975  5983
NCYL2   005524  3510#  5596*  5599  5604*  5609*
NED   = 010000  3041#  6440  9247  9257  9827  9905  10248  10618
NEDHDL  025550  6440#  6977
NEM   = 004000  3040#  9242  9842  10008  10248
NEWDRV  016754  5225#  5235  6396  8982  9496
NEWMSG  007564  4136#  6588
NEWON   005540  3516#  6452*  6978*  6982
NEWPAS  016742  4435  4520  5222#  6433
NGOFST  005556  3522#
NMTSTS= 000021  3601#  5350  5374  5400  5448  5487  5535  5591  5627  5651  5684  5734  5875
                5957  6029  6106  6189  6264
NOCHK = 000400  3227#  10482  10544  10585  10612  10630  10651  10674  10683  10717
NOCLKS  011440  4317#  4542
NODRTS  010003  4164#  4663
NODSC = 004000  3232#  9458  10070
NOFALS  011431  4313#  4662  8592
NORTRY  003143  3449#  6893*  9519
NOTALD  010701  4248#  4937  4942
NOTDFT  016236  5059  5076#
NOTMSG  063520  4500  12371#
NOTYPE  003147  3454#  4498  4501*
NTREDY  007636  4145#  7001
NULINP  015042  4839  4854#
NXDRIV  007622  4143#  6987
NXF   = 000004  3052#
OCTBIN  052430  4884  5095  5175  5215  6603  7332  10748#
OFFSET= 000115  2980#  10527
OFINUS  005526  3511#
OFST  = 000004  3072#
ONECYL  011605  4335#  5935
ONLINE  005536  3515#
OPCOMP  003122  3432#
OPCONT  003131  3439#
OPI   = 020000  3064#  8566  9359  9848  10014  10250
OPIERR= 000010  4081#  9361  9514
OPNLST  016126  4924  5052#  5057  5064
OR    = 000200  3036#
```

```
OREQSW= 000020           3459#
OVLODR  007404           4114#  4967
O.WAIT  003054           3376#  9748    9752*   9862*   9866    9882*   9948*   9961*   9976*   10520*  10580*  10642*  10659*
PACK  = 000103           2975#  7012
PARMDE  010567           4234#  4908
PARM0   002630           3265#  3420    6892    7053    8711    8724    9471    9540    9591    9625
PARM1   002714           3297#  6898*   6901*   9215    9575    9609
PART  = 000000           1766#  1779    1782    1785    1794    1806    1814    1820    1821    1984    1986    1988    1990
                         2003   2025    3480    3488    3529    3560    3577    3582    3599    3608    3611    3616    3618
                         3623   3630    3633    3638    3640    3645    3655    3661    3671    3675    3683    3685    3694
                         3697   3702    3704    3709    4093    4095    4316    4398    4438    4495    4498    4525    4944
                         4983   5319    5339    6385    6435    6733    6824    6826    7044    7513    7591    8318    8488
                         8497   8838    12369   12370   12418   12454   12455
PART1   007344           4108#  4497
PART2   007350           4109#
PAT   = 000020           3090#
PATRN   005544           3518#  6272*   8120    8191    8298    8717
PAT00   006216           3715#  8122    8137    8193    8208
PAT01   006256           3737#
PAT02   006316           3759#
PAT03   006356           3781#
PAT04   006416           3803#
PAT05   006456           3825#
PAT06   006516           3846#
PAT07   006556           3868#
PAT08   006616           3889#
PAT09   006656           3911#
PAT10   006716           3933#
PAT11   006756           3955#
PAT12   007016           3977#
PAT13   007056           3999#
PAT14   007116           4021#  8049
PAT15   007156           4041#  7259    7337*   7343*
PBLKT   003112           3420#  9745    9832    9858    10026   10051   10469*
PBSVAL= 001000           3228#  10407
PCA   = 004000           3097#
PCD   = 010000           3098#
PCLKF   003132           3440#  4526*   4530*   7596    7617
PCVEC   003162           3467#  4531
PFIFTN  011057           4269#  5022
PGE   = 002000           3039#  9305    10248
PIP   = 020000           3082#
PIRQ  = 177772           1837#
PIRQVE= 000240           1931#
PKRB    003156           3465#
PKS     003152           3463#  4529    7604*   7621*
PKSB    003154           3464#  7600*   7603*
PLOFST  005554           3521#
PMA     005602           3529#  6289*   6290*   6686    6688    8139    8210    8290    8292    8691*   8692*   8693*   8694
                         8695   8715*   8716*   8719*   8720*
PRDFLT  005776           3627#  5015    5066    5069    5108    5149    5246*   5269*
PREPAR  026532           6295   6684#   8700    8758
PREPKB  025766           4812   4996    5008    5033    5353    5377    5403    5451    5490    5538    5594    5630    5654
                         5687   5737    5878    5960    6032    6109    6192    6267    6505#   8465    8899
PRMBUF  012424           4419#  7226*   7227
PRMINP  007441           4119#  4573
```

```
PRMLIM  006042          3650#  5243*  5247*  5248*  5266*  5270*  5271*  7390   7392   7400   7403   7405   7408
PRMLST  005732          3605#  5014   5067   5107   5148   7228   7232   7395*  7411*  7412*
PRMNEM  006152          3691#  5009   5088   5144   7226   7230   7286   7383
PRMPHO  005264          3495#  8926*  12104
PRMPLO  005262          3494#  8927*  12104
PRMPSP  012445          4423#  5129
PROMPT  012442          4422#  4695   4859   4909   5080   7309
PRVCMD  005232          3490#  4514   8884*  8886*  8915   9182
PR0    = 000000         1854#  4513   5225   7056   9541   9613
PR1    = 000040         1855#
PR2    = 000100         1856#
PR3    = 000140         1857#
PR4    = 000200         1858#  4506
PR5    = 000240         1859#  3352
PR6    = 000300         1860#
PR7    = 000340         1861#  4445   4511   4533   6667   11372  11373
PS     = 177776         1834#  1835   4445*  4513*  5225*  7056*  9734   9740*  9754*  10463  10464*  10723*
PSTART  012522          1987   4442#
PSW    = 177776         1835#  9541*  9613*
PT      005760          3617#  5017   6272   6273   6299   6340   6751   7289
PWRMSG  055052         11375   11379#
PWRVEC= 000024           1926#  4460*  4461*  11363*  11371*  11372*
P.ASOF= 000032           3249#  9169   10076*  10077   10110*  10341*  10529*  10530
P.A00  = 000040          3252#  6996   7004   9172   9581   10067*  10068   10084*  10085   10398*  10606*
P.A01  = 000044          3254#  9174   10377*
P.A10  = 000050          3256#  9176   10384*
P.A11  = 000054          3258#  7463   9178   10391*
P.BAHI= 000007           3211#  6288*  6710*  8697*  9158
P.BALO= 000010           3212#  6117*  6197*  6276*  6285*  6709*  7023*  7500*  8484*  8534*  8546*  8694*  9161
                        10563  10660
P.BAR  = 000024          3246#  9168   10107*  10337*
P.B00  = 000042          3253#  9173   10062*  10063   10399*  10607*
P.B01  = 000046          3255#  9175   10378*
P.B10  = 000052          3257#  9177   9603   10385*
P.B11  = 000056          3259#  9179   9983*  10392*
P.CMND= 000001           3205#  5288*  5355*  5749*  5886*  5968*  6040*  6110*  6139*  6142*  6193*  6218*  6318*
                         6322*  6330*  6337*  6895*  6965*  6976*  7012*  7014*  7016*  7054*  7459*  7497*  7525*
                         7528*  7762*  7779*  8032*  8448*  8487*  8490*  8525*  8885*  9149   9484*  9531*  9576*
                         9612*  9615*  9878   10495   10499   10511   10521   10527   10533   10536   10542   10567   10574
                        10593  10640  10657  10679  10688
P.CS1  = 000016          3243#  5757*  5777*  5778   6916   6966*  8887*  9162   9280   9298   9644   9677   9899*
                         9951*  9989*  9990*  9992*  9993   10102*  10104*  10331*  10334*  10485  10539*  10540*  10542*
                        10546*  10547  10571*  10574*  10576*  10577*  10587*  10588  10596*  10597*  10599*  10600  10602*
                        10603  10608  10647*  10648*  10650*  10653*  10654  10670*  10671*  10673*  10676  10695*  10696
                        10705*  10706
P.CS1H= 000007           3210#  5748*  5756*  5776*  6899*  6900*  6901*  7025*  7026*  7493*  8524*  9623*  9624*
                         9989   10230  10539  10576  10596  10647  10661  10670
P.CS2  = 000020          3244#  6440   9163   9237   9242   9247   9249   9255   9257   9305   9355   9439   9905
                        10105*  10335*  10493*  10518  10583*  10584  10595  10614  10618  10644*  10645  10669
P.CYLN= 000002           3206#  5358*  5379*  5382*  5405*  5408*  5410*  5411   5414*  5416   5453*  5456*  5460*
                         5464*  5465*  5469*  5493*  5514*  5541*  5543*  5564*  5566*  5597*  5599*  5632*  5655*
                         5657*  5659*  5693*  5695*  5711*  5713*  5887*  5890*  5907   5910*  5927   5969*  5975*
                         5979*  6041*  6045*  6063*  6111*  6194*  6277*  6297*  6351*  6352   6367*  6368   6371*
                         6372   7019*  7021*  7498*  7531   7534   8099   8331   8336   8365   8377   8388   8526*
                         8628   8682*  9151   9611*  9987   10523  10565  10667
P.DCYL= 000030           3248#  8632   8678   9164   9412   9634   9666   10109*  10343*
```

```
P.DRVN= 000000  3204#  6897*  6944*  9875  10052  10228  10467
P.DS  = 000036  3251#  9170  9290  9294  9345  10112*  10339*  10620
P.DTS = 000026  3247#  8634  8636  8679  8680  9165  9413  9415  9602  9637  9638  9667  9673
                10108*  10338*
P.EPAT= 000062  3261#  9436  9901*  10487
P.EPOS= 000060  3260#  9435  9902*
P.ER  = 000034  3250#  8566  9171  9236  10111*  10340*
P.OFST= 000006  3209#  10529
P.PRST= 000014  3214#  9263  9276  9448  9453  9458  9463  9578  9747*  9860*  9894*  9922*  9929
                9933*  10030*  10053  10057*  10070*  10088*  10098*  10124  10128*  10138*  10150*  10255*  10407*
                10482*  10519*  10543*  10544  10569  10575*  10581  10585  10612  10622*  10630  10651  10674
                10683  10717
P.SECT= 000004  3207#  6116*  6125  6129*  6130  6134*  6136  6147  6171*  6172  6196*  6280*  6305*
                6343*  6344  6356*  6357  7496*  8026  8037*  8076  8355  8531*  8540*  8541  8547*
                8557*  8558  8630  8684*  9154  9610*  9988  10524  10566  10668
P.TRCK= 000005  3208#  6112*  6195*  6200  6204*  6205  6214*  6215  6221*  6222  6278*  6304*  6347*
                6348  6359  6361*  7022*  7499*  8027*  8029  8034*  8035  8090  8338  8368  8527*
                8629  8683*  9152
P.WC  = 000012  3213#  6118*  6198*  6279*  6306*  6307*  7024*  7501*  8396  8409*  8485*  8502  8533*
                8687*  8713  8750  9156  10564
P.WCR = 000022  3245#  9166  10106*  10336*
QNEWSW= 000000  2964
Q.INIT= 040000  3235#
RCDASW= 000010  3458#
RCLREQ= 004000  4089#  9283  9288  9333  9338  9343  9348  9482
RDALHD= 000164  2993#  9878  10657
RDCHR = 104406  7121  11674#
RDCHRS  030560  4630  4682  4697  4790  4834  4861  4910  4968  5054  5082  5131  5203  6589
                7115#  7293  7311
RDDATA= 000121  2982#  6142  6330  7016  7497  8525
RDGATE= 100000  3101#
RDHDO   044466  9370  9395  9600#
RDHEAD= 000125  2984#  5288  5749  7528  9615  9992  10673
RDSTAT= 000141  2992#  6976  7459  9576  10593
RDY   = 000200  3007#  9478  10550  10603
RECAL = 000113  2979#  5355  7014  8448  9484  10536
RECODE  005504  3502#  6148  6970*  7028  7051*  8253  8743  8746  8824*  8832*  8883*  8951  8961*
                9212*  9213  9240*  9245*  9266  9268  9283*  9288*  9328*  9333*  9338*  9343*  9348*
                9353*  9361*  9365  9371  9386*  9390  9396  9411*  9418  9421*  9425*  9428*  9441*
                9442*  9451*  9456*  9461*  9466*  9474  9480*  9482  9488  9490*  9514  9516*  9521
                9528*  9548*  9580*
REDBSF  036630  5307  8522#
REISSU  003140  3446#  6315*  9107*
RELEAS= 000140  2991#  6337  10640
REPLPK  007714  4154#  4680
REPSUP  041650  5764  6662  6968  7533  7784  8190  8543  8889  9146#  9218
RESREG= 104410  5768  5817  6573  6654  6711  6756  6837  7424  7482  7695  7817  7837  7887
                8014  8038  8055  8102  8171  8252  8313  8411  8415  8492  8513  8618  8643
                8701  8729  8825  8906  8938  9078  9197  9543  9592  9626  9685  11020  11072
                11676#
RESVEC= 000010  1921#
RETANL  044270  9216  9485  9547#
RETNML  044306  6455  8570  9217  9550#
REVRSE  011701  4347#  6010  7919
RHEDHD  022350  5754  5854#
RKASOF= 000016  2960#  8892  9569  9888  9958  10038  10293  10341  10530*  10555
```

```
RKBA   = 000004  2955#  9568   10289  10337  10563*
RKBADR  007467   4123#  4577
RKBAS   003036   3350#  4558*  4562*  4576   4579*  6891   7782   9373   9398   9477   9530   9561   9741
                 9810   10474
RKCS1  = 000000  2953#  4691*  5778*  5855   6992*  7765   7783*  9374*  9378   9399*  9403   9563   9833
                 9885   9943   9993*  10002  10184* 10185  10194* 10233* 10234  10258* 10286  10547* 10549
                 10588* 10600* 10602  10632* 10654* 10676* 10685* 10695  10705  10720*
RKCS2  = 000010  2957#  6954*  6963   8867*  8871   8894*  9564   9811   9971   10229* 10287  10335  10518*
                 10584* 10595* 10645* 10669* 10694*
RKDA   = 000006  2956#  9566   9988*  10290  10338  10524* 10566* 10668*
RKDB   = 000024  2963#  5299   5300   7530   9375   9376   9377   9400   9401   9402   9967   9968   9969
RKDC   = 000020  2961#  9565
RKDCYL= 000020  2962#  9987*  10294  10343  10523* 10565* 10667*
RKDS   = 000012  2958#  8873   9478   9570   9836   10291  10339
RKECPS= 000030  2968#  9902   10298
RKECPT= 000032  2970#  9901   10299
RKER   = 000014  2959#  6972   9571   9845   10011  10292  10340
RKMR1  = 000026  2964#  9979*  10058* 10295  10372* 10379* 10386* 10393* 10494*
RKMR2  = 000034  2965#  10296  10606
RKMR3  = 000036  2966#  10297  10607
RKPAT  = 000032  2969#
RKPOS  = 000030  2967#
RKPRI   003042   3352#  4509   4567*  4586   4602*  9740   10464
RKPRTY  007535   4131#  4591
RKVADR  007512   4127#  4582
RKVEC   003040   3351#  4507   4563*  4564*  4581   4584*  5754*  5848*  7761*  7776*
RKWC   = 000002  2954#  9567   10288  10336  10564*
RLS    = 000010  3031#  10644
RNDADR  032510   5631    7581#
RNDLNG= 016514   4076#
RNDSHT= 002734   4075#
ROTIMS  011540   4328#  5832
RUNTST  016646   4964   5200#
RVTERM  033514   5982   7826#
RWBUF   063520   3618   3640   3675   6117   6144   6197   6276   7023   7031   7500   7505   8010   8012
                 8124   8195   8478   8484   8715   12369#
RWDTST  024444   5322   6227#
RWTINX= 000040   6228#  6749
R.ABNL  050466   9753   9863   9938   9955   10133  10153  10413# 10623
R.CONT  050512   9815   9853   9917   9926   9997   10019  10042  10048  10081  10121  10145  10190  10242
                 10265  10403  10421# 10637  10691  10700  10713
R.NORM  050500   9891   9984   10093  10417# 10560  10633  10721
SAVPAR  003206   3478#  6698*  8143   8214
SAVPRS  005560   3523#  8712   8723
SAVREG= 104407   5782   6534   6640   6684   6744   6816   6819   7380   7457   7640   7806   7826   7870
                 8009   8025   8047   8070   8118   8185   8247   8333   8476   8500   8580   8627   8676
                 8710   8722   8814   8858   8914   8973   9091   9147   9209   9560   9601   9665   11005
                 11046  11675#
SAVWRD  003210   3479#
SAVXDP  027202   6783   6816#
SCLR   = 000040  3034#  10694
SCNDRV  027604   4758   6942#
SCOPER= 104411   6339   11677#
SCOPE1  055074   11390# 11677
SCRACH  005542   3517#  4652*  4653*  4654   5691*  5697*  5709*  5715*  5774*  5819   5972*  5974*  6043*
                 6080*  6081   6146*  6152   6157*  6160   6960*  6961*  7156*  7158*  7159   7169*  7170*
```

```
                    7171    7810    7830    8189*   8255    8268*   8271
SECNL1  010670      4246#   4932
SECNL2  010720      4251#   4936
SEEK  = 000117      2981#   5886    5968    6040    7525    7762    10511   10521
SEEKER  032330      5359    5380    5383    5406    5409    5415    5454    5457    5461    5466    5470    5494    5515
                    5542    5544    5565    5567    5598    5600    5633    5656    5658    5660    5694    5696    5712
                    5714    7524#
SEEKHD  033426      7761    7793#
SELDRV= 000101      2974#   6965    8884    10650
SELECT  005534      3514#
SELP15  011106      4273#   7305
SERNM   011351      4304#   7462    7504
SETPRM  016412      4927    5128#   5134    5139    5183
SETTUP  030416      5348    5372    5398    5446    5485    5533    5589    5625    5649    5682    5732    5873    5955
                    6027    6104    6133    6187    6209    6213    6262    6312    7049#
SIZMEM  026020      4544    6534#
SKI   = 000002      3050#   9330
SM      005774      3623#   5689
SOFTBS  011377      4308#   8616
SPACE1  012421      4417#   4858    5079    7200    7308
SPACE2  012420      4416#   8597    8603    9020    9024    9027    9030    9050    9055    9070
SPACE3  012417      4415#
SPACE4  012416      4414#
SPACE6  012414      4413#   7207
SPAR  = 020000      3014#   9280    10246
SPCDMX  012225      4388#   6008    6013
SPDLSS= 000020      3074#   9294
SRHTBS  040004      8775#   8816
SRTSPL= 000111      2978#   10533
SR0   = 177572      1940#   6653*   6663    6861*   6863*   6865*   6867*   8144*   8159*   8215*   8311*   11545   11560*
                    11567*
SR1   = 177574      1941#   6664
SR2   = 177576      1942#   6665
SR3   = 172516      1943#   6380*   6652*   6668    11557*
ST      005772      3622#   5305    5719
STACK = 001100      1825#   4452    4627    5226    7049*   7050    8447    11374
STALL   032426      7552#   8859
STALLS  005512      3505#   4519*   4628*   4907*   5305*   5692*   5710*   5719*   7561
STKLMT= 177774      1836#
STPASS  016730      5074    5220#
STRCMD  040542      5758    7766    8863    8914#
SUBCLR= 000177      2995#   6895    7054    7779    9531    9612    10688
SUBVER  007321      4104#   4496
SUMHI1  003200      3475#   5786*   5787*   5825*   5894*   5895*   5988    5994*   6049*   6050*   7417*   7643*   7665*
                    7666*   7674*   7687    7732*   7816*   7874    7878*   8335*   8385
SUMHI2  003204      3477#   5914*   5915*   5990    6002*   6067*   6068*   7734*   7813*   7882    7885*
SUMLO1  003176      3474#   5785*   5826*   5845    5893*   5987    5995*   6006    6048*   7416*   7418    7642*   7664*
                    7675*   7688    7731*   7875    7879*   7915    8334*   8361*   8373*   8383
SUMLO2  003202      3476#   5913*   5989    6003*   6011    6066*   7733*   7815*   7883    7886*   7930
SVAL  = 100000      3084#
SVPRMS  037562      6313    8710#   8755
SWR     001140      2055#   4450    4472*   4474    4480*   4487*   6158    6166    6269    6316    6320    6326    6328
                    6332    6620    7526    7554    8269    8296    8983    9010    9038    11280   11287   11299   11301
                    11392   11415   11433   11440
SWREG   000176      1981#   4480    6620    6622    6625*
SWRMSG  007555      4134#   6623
```

```
SW0    = 000001            1889#
SW00   = 000001            1879#   1889
SW01   = 000002            1878#   1888
SW02   = 000004            1877#   1887
SW03   = 000010            1876#   1886
SW04   = 000020            1875#   1885
SW05   = 000040            1874#   1884
SW06   = 000100            1873#   1883
SW07   = 000200            1872#   1882
SW08   = 000400            1871#   1881
SW09   = 001000            1870#   1880
SW1    = 000002            1888#
SW10   = 002000            1869#
SW11   = 004000            1868#
SW12   = 010000            1867#
SW13   = 020000            1866#   8983
SW14   = 040000            1865#
SW15   = 100000            1864#
SW2    = 000004            1887#
SW3    = 000010            1886#
SW4    = 000020            1885#
SW5    = 000040            1884#
SW6    = 000100            1883#
SW7    = 000200            1882#
SW8    = 000400            1881#
SW9    = 001000            1880#
S.ACLO= 000100             3119#
S.BRHM= 000100             3134#
S.BRKE= 040000             3156#
S.CART= 000400             3136#
S.DCLO= 010000             3126#
S.DIB  = 002000            3152#
S.DOOR= 000200             3135#
S.DRA  = 000040            3105#
S.DROT= 020000             3127#
S.DRY  = 000200            3107#   6996
S.DSC  = 040000            3114#   9920    10068   10085   10148
S.FLT  = 000200            3120#   10063
S.FORM= 001000             3109#
S.FWD  = 002000            3138#
S.HDFL= 000200             3149#
S.HDHM= 000040             3133#
S.ICYL= 000040             3118#
S.ILF  = 000400            3121#
S.LIMD= 020000             3155#
S.LOAD= 010000             3140#
S.MHD  = 000400            3150#
S.NMOV= 010000             3154#
S.OFF  = 002000            3110#
S.PAR  = 001000            3122#   9923    10400
S.PIP  = 020000            3113#   9931    10126
S.PLO  = 004000            3153#
S REV  = 004000            3139#
S.RTZ  = 020000            3141#
S.SECT= 000020             3146#
S.SKI  = 002000            3123#
```

```
S.SPIN= 010000  3112#
S.SPLS= 010000  3125#
S.SPOK= 001000  3137#
S.TYPE= 000400  3108#
S.UNLD= 040000  3142#
S.UNS = 040000  3128#
S.VV  = 000100  3106#
S.WCLK= 000040  3147#
S.WGAT= 000100  3148#
S.WLE = 004000  3124#
S.WRL = 004000  3111#  7004
S.XDOK= 000020  3132#
S.XERR= 001000  3151#
S0      005746  3612#  4930  5303
S1      005750  3613#  4930  5304
S2      005752  3614#  4934  5297
S3      005754  3615#  4934  5298
TBITVE= 000014  1922#
TCONHI  003166  3470#  7690*  7948
TCONLO  003164  3469#  7692*  7949
TELDRV  013740  4635  4643#  4700  4711  4766
TIMER   033104  5779  7661  7711#  7716  7769
TIMSEK  033246  5891  5911  5976  5980  6046  6064  7757#
TIMSKP  011454  4319#  7653
TKVEC = 000060  1929#  4505*  4506*
TLSTHD  010411  4213#  4810  4855
TOPROC  044316  5765  7785  9094  9265  9559#
TPVEC = 000064  1930#
TRACK   005530  3512#
TRAPVE= 000034  1928#  4458*  4459*  11373*
TRKCHK  034262  6210  6220  8025#
TRKNLW  010732  4253#  4941
TRNSFR  037674  6319  6323  6331  8739#
TRTVEC= 000014  1923#
TSTDRN  011312  4297#  5310
TSTING  003120  3430#  4512*  5221*  5745*  5751*  8430
TSTLST  005626  3559#  3601  4826  4847  4869  4889*  4894  4898*  6748  7084  7208
TSTMDS  010205  4187#  4788
TSTMSG  056306  9004  11679#
TST1    017730  5320  5346#
TST10   021154  5591  5623#
TST11   021226  5627  5647#
TST12   021314  5651  5680#
TST13   021554  5684  5730#
TST14   022366  5734  5743  5871#
TST15   022706  5875  5884  5892  5912  5953#
TST16   023244  5957  5966  5977  5981  6025#
TST17   023570  6029  6038  6047  6065  6102#
TST2    020010  5350  5370#
TST20   024210  6106  6185#
TST21   024444  6189  6260#
TST3    020070  5374  5396#
TST4    020176  5400  5444#
TST5    020342  5448  5483#
TST6    020546  5487  5531#
TST7    020776  5535  5587#
```

```
TWOTOS= 002000  4088#  9266  9580
TYPAVG  034214  5846  6007  6012  7916  7931  7995#
TYPBSF  037120  5317  8580#
TYPDS = 104405  5836  5842  6411  7911  7926  11671#
TYPE  = 104401  4495  4496  4497  4500  4504  4542  4552  4573  4577  4582  4591  4603  4605
                4606  4629  4639  4640  4643  4651  4654  4662  4663  4672  4680  4690  4695
                4709  4710  4788  4789  4796  4797  4810  4814  4833  4842  4843  4855  4858
                4859  4901  4902  4906  4908  4909  4916  4917  4932  4936  4937  4941  4942
                4959  4967  4974  4975  4998  5022  5035  5053  5062  5063  5079  5080  5111
                5112  5129  5137  5138  5202  5211  5212  5310  5832  5837  5838  5843  5844
                5847  5935  59R5  6005  6008  6010  6013  6089  6409  6412  6473  6479  6483
                6487  6489  54    6521  6522  6567  6571  6572  6588  6598  6599  6623  6951
                6986  6987  7000  7001  7008  7009  7030  7035  7036  7126  7132  7138  7144
                7155  7159  7163  7171  7176  7178  7179  7181  7200  7207  7227  7252  7258
                7292  7301  7302  7305  7308  7309  7347  7348  7415  7421  7461  7462  7481
                7503  7504  7508  7652  7653  7904  7907  7913  7917  7919  7922  7928  7932
                7969  7973  7982  7986  7995  7999  8458  8459  8583  8584  8585  8586  8592
                8593  8597  8603  8607  8616  8981  8997  8998  8999  9003  9004  9009  9012
                9013  9014  9015  9020  9023  9024  9025  9026  9027  9030  9033  9034  9036
                9050  9055  9059  9065  9070  9071  9073  10866  11113  11178  11253  11282  11290
                11375  11667#
TYPERR  040732  8973#  11289
TYPFMT  003124  3434#  4762  5240*  5263*  6900  6942*  6955  6974*  7017  7026
TYPLST  015654  4921  4994#
TYPMAX  034170  5840  7909  7924  7982#
TYPMIN  034144  5834  7906  7921  7969#
TYPOC = 104402  8596  8602  8606  9019  9029  9032  9052  11668#
TYPON = 104404  11670#
TYPOS = 104403  6585  7204  7209  7255  7260  9006  11669#
TYPPAT  031206  5034  7251#  7307
TYPPRM  031126  4997  5078  7226#
TYPTMS  033724  5937  6091  7902#
TYPTST  031072  4813  4857  7200#
T.ASOF  003014  3337#  9888*  9889  9913  9958*  9959  10038*  10039  10045  10076  10110  10118  10142
                10293*  10555*  10556
T.BA    003006  3334#  10107  10289*
T.CS1   003000  3329#  9833*  9885*  9886  9899  9910*  9943*  9945  9951  9980*  10002*  10004  10034
                10059*  10073*  10103*  10104  10115*  10139*  10185*  10186  10226  10230*  10231*  10233  10237
                10246  10286*  10333*  10334  10374*  10381*  10388*  10395*  10549*  10550  10552
T.CS2   003002  3331#  9811*  9812  9827  9830  9842  9871  9971*  9972  10008  10022  10105  10227*
                10228*  10229  10239  10248  10287*
T.DA    003010  3335#  10108  10290*
T.DB    003034  3346#
T.DC    003012  3336#  10109  10294*
T.DS    003020  3340#  9836*  9837  10112  10291*
T.ER    003016  3339#  9845*  9848  10011*  10014  10111  10250  10292*
T.MR1   003022  3341#  10295*
T.MR2   003024  3342#  9920  9931  10067  10084  10126  10148  10296*  10377  10384  10391  10398
T.MR3   003026  3343#  9923  9983  10062  10297*  10378  10385  10392  10399  10400
T.PAT   003032  3345#  10295*
T.POS   003030  3344#  10298*
T.WCR   003004  3333#  10106  10288*
UBMPRS  003144  3450#  6378  6565*  6650  6699  8257  8924
UEXATT= 000010  3221#  9463  9894
UFE   = 000400  3037#  9249  9255  9827  9905  10248  10618
UNLOAD= 000107  2977#  10499
```

```
UNS    = 040000          3065#   9325
UPE    = 020000          3042#   9237    9842   10008   10248
VFHDSW= 000002           3456#
VV     = 000100          3077#
WC       005766          3620#   4945    6306    6758
WCDASW= 000004           3457#
WCE    = 040000          3043#   9439    9842   10008   10248
WCEFLG   003141          3447#   6324    8536*   8538    8553*   8555    8569*   8741*   8745*
WCERR  = 000100          4084#   8743    9442    9514
WC2BIG   010744          4255#   4959
WDSXFR   005572          3524#   8119    8188    8244    8642*   8685    8713*   8714*
WHDSW  = 000001          3455#
WLE    = 004000          3062#   9321    9848   10014
WORDSP   012430          4420#   7252
WRDATA= 000123           2983#   6110    6193    6218    6318    8487   10571
WRHEAD= 000127           2985#
WRL    = 004000          3081#
WRTCHK= 000131           2986#   6139    6322    8032    8490   10567
WRTGAT= 040000           3100#
WRTLOK   007650          4147#   7009
WRTSEC   036476          8476#
W.DRV    003114          3426#   9742*   9857*   9883*   9935*   9950*   9963*   9977*  10025*  10092*  10101*  10130*  10152*
                        10423*  10472*  10535*  10538*  10559*  10611*  10629*
W.MILI   003060          3379#   9737   10703
W.MIN    003066          3395#  10535
W.MTIM   003056          3378#   9735*   9737*
W.SEC    003062          3392#  10472
W.TIME   003100          3401#   9738    9744*   9856*   9884*   9934*   9949*   9962*   9978*  10024*  10091*  10100*  10129*
                        10151*  10422*  10471*  10558*  10610*  10628*
W.WCK  = 000200          3226#  10482   10543   10569   10575
W.WTCH   045104          8903    9730#  10548   10554   10601
W.8SEC   003064          3394#   9935   10130   10538
XDPSAV   005550          3520#   6781*   6782*   6796*   6797*   6828    6829    6855
XDPSVD   003135          3443#   4545    4548*   6767*   6777*   6800*   6821    8445*
XDP1     027230          6818    6824#
XDP2     027276          6822    6825    6837#   6844    6875
XDP3     027322          6831    6846#
XDP4     027302          6833    6839#
XOVLAD   003134          3442#   6268*   6291*   6375    6820*   8439
XXDPAD   005546          3519#   6722*   6726*   6727*   6728*   6769    6775    6827
XXDPCH   003117          3429#
XXDPMG   007240          4095#   4552
$APTHD   001000          2012    2018#
$ASTAT= ****** U        11491   11506
$ATYC    055402         11462   11464#
$ATY1    055356         11460#
$ATY3    055364         10851   11461#
$ATY4    055374         11295   11463#
$AUTOB   001134          2052#
$BASE    001374          2159#   4558
$BDADR   001122          2047#
$BDDAT   001126          2049#
$BELL    001310          2110#   11282   11314
$CHARC   053144         10868*  10878*  10885   10894*  10899#
$CKSUM= ****** U        11674
$CMTAG   001100          2035#   4447    4448    4456    4462    4463    4464
```

```
$CM1   = 000040          2067#  2068#  2069#  2070#  2071#  2072#  2073#  2074#  2075#  2076#  2077#  2078#  2079#
                         2080#  2081#  2082#  2083#  2084#  2085#  2086#  2087#  2088#  2089#  2090#  2091#  2092#
                         2093#  2094#  2095#  2096#  2097#  2098#  2099#
$CM2   = 000100          2067#  2068#  2069#  2070#  2071#  2072#  2073#  2074#  2075#  2076#  2077#  2078#  2079#
                         2080#  2081#  2082#  2083#  2084#  2085#  2086#  2087#  2088#  2089#  2090#  2091#  2092#
                         2093#  2094#  2095#  2096#  2097#  2098#  2099#
$CM3   = 000040          2065#  2067
$CM4   = 000011          2099#  2100#  2101#  2102#  2103#  2104#  2105#  2106#  2107#  2108#
$CNTLG  054735          11352#
$CNTLU  054730          11351#
$CORE   056032          11543  11570#
$CPUOP  001346           2133#
$CRLF   001315           2112#  4603   4672   4814   4998   5035   5847   6490   6571   6572   7176   7181   7421
                         7481   7508   7907   7917   7922   7932   8586   8593   8607   9003   9013   9015   9023
                         9026   9036   9059   9065   9073  10867  10902  11290  11314
$CROUT  056062          11570  11577#
$DB_K   054404          11219  11253  11261#
$DB2D   053506           7971   7984   7997  11046#
$DB2O   053366           6569   7234   7419   7506  11005#
$DECVL  053666          11048  11094#
$DEVCT  001330           2124#  5223*  6395*
$DEVM   001376           2160#  4740
$DOAGN  025472           6405   6414   6420#
$DTBL   054374          11222  11257#
$ENDAD  025462           1998   6416#  11309
$ENDCT  025430           4462   6407#
$ENDMG  025501           6409   6424#
$ENULL  025476           6412   6423#
$ENV    001340           2129#  6427   10846  11292  11469  11493
$ENVM   001341           2130#  4485    4559   4741   6988  10848  10853  11471
$EOP    025356           6264   6393#
$EOPCT  025422           4462*  5073*   5217*  6404#  6408
$ERFLG  001103           2038#  11277*  11314  11390  11404  11429  11431  11437*  11457
$ERMAX  001115           2044#   4465*   4492*  7072*  11431  11452*  11457
$ERROR  054414           4456   11276#
$ERRPC  001116           2045#  11284*  11285*  11286  11314  12089
$ERRTB  001400           2177#   8994
$ERTTL  001112           2042#  11283*  11314
$ESCAP  001306           2109#   4464*  11305  11307  11314  11451*
$ETABL  001340           2128#
$ETEND  001400           2024    2161#
$FATAL  001322           2121#  11497*
$FFLG   055622          11460*  11463*  11491  11500*  11508#
$FILLC  001156           2063#  10871   10902
$FILLS  001155           2062#  10902
$GDADR  001120           2046#
$GDDAT  001124           2048#
$GET42  025452           6413#
$GTSWR= ****** U        11673
$HD    = 000000          1794
$HIBTS  001000           2019#
$HINUM  052662           4523*   8052   10803  10811  10816*  10821#
$HIOCT  052562           5098    5178    6606   7335  10773*  10786#
$ICNT   001104           2039#  11442*  11443  11445*  11456
$INTAG  001135           2053#
$ITEMB  001114           2043#   8989   11286*  11294  11314
```

```
$KTNEX  056024          11544   11569#
$KTOUT  056014          11561   11566#
$KT11   055662           6292    6535*   6537    6830    8140    8211    8300    8309    8698    8756   11542#  11546*  11569*
$LF     001316           2113#  10902   11314
$LFLG   055621          11501*  11507#
$LONUM  052664           4524*   7557    7582    8051   10802   10809   10815*  10822#
$LPADR  001106           2040*   4466*  11377   11435*  11449*  11454   11456
$LPERR  001110           2041#   4467*   6132*   6208*   6212*   6311*  11304   11394   11435   11450*  11456
$LSTAD  056126           6539    6552   11582*  11589#
$LSTBK  056130           6543   11583*  11590#
$MADR1  001352           2146#
$MADR2  001356           2150#
$MADR3  001362           2153#
$MADR4  001366           2156#
$MAIL   001320           2020    2024    2119#   4484    5347    5371    5397    5445    5484    5532    5588    5624    5648
                         5681    5731    5872    5954    6026    6103    6186    6261   10846   11292   11448
$MAMS1  001350           2140#
$MAMS2  001354           2148#
$MAMS3  001360           2151#
$MAMS4  001364           2154#
$MBADR  001002           2020#
$MFLG   055620          11461*  11467   11502*  11506#
$MNEW   054753          11355#
$MSGAD  001334           2126#  11477*  11480
$MSGLG  001336           2127#  11482*
$MSGTY  001320           2120#  11475   11483*  11495   11499*
$MSWR   054742          11353#
$MTYP1  001351           2141#
$MTYP2  001355           2149#
$MTYP3  001361           2152#
$MTYP4  001365           2155#
$MXCNT  055354          11446   11456#
$NULL   001154           2061#  10873   10902
$NWTST= 000001           5339#   5341    5363#   5365    5387#   5389    5421#   5423    5474#   5476    5520#   5522    5572#
                         5574    5616#   5618    5637#   5639    5664#   5666    5723#   5725    5862#   5864    5941#   5943
                         6017#   6019    6095#   6097    6177#   6179    6229#   6231
$OCNT   054*54          11150*  11179*  11192#
$OCTVL  053470           7464   11007   11032#
$OMODE  054166          11145*  11149*  11154   11157*  11168*  11194#
$OVER   055340          11416   11436   11444   11453#
$PASS   001326           2123#   4484*   5220*   5311    6401*   6402*   6410    6423    6429    7077
$PASTM  001006           2022#
$PWRCT  055050          11368*  11369*  11378#
$PWRDN  054764           4460   11363#  11371
$PWRUP  054776          11363   11368#
$QUES   001314           2111#   4606    4640    4710    4797    4843    4902    4917    4975    5063    5112    5138    5212
                         6522    6599    7179    7302    7348   10902   11314
$RAND   052564           7556    7581    8050   10798#
$RDCHR  054610          11330#  11674
$RDDEC= ****** U        11675
$RDLIN= ****** U        11675
$RDOCT= ****** U        11675
$RDSZ = 000000          11351#
$REGAD  001160           2065#
$REG0   001162           2067#   9150   12089
$REG1   001164           2068#  12089
```

```
$REG10  001202          2075#   6162*   6668*   8243*   8244*   8245    8283*   8286    9375*   9400*   9435*   12093   12101
$REG11  001204          2076#   6163*   8284*   9376*   9401*   9436*   12093   12101
$REG12  001206          2077#   6164*   8285*   9377*   9402*   9429*   9431*   12093   12101
$REG13  001210          2078#   8287*   8289*   8291*   8292*   12093   12101
$REG14  001212          2079#   8286*   8288*   8290*   12096   12101
$REG15  001214          2080#   8293*   8294*   12096   12101
$REG16  001216          2081#   8222*   12097   12101
$REG17  001220          2082#   12097
$REG2   001166          2069#   12089
$REG20  001222          2083#   12097
$REG21  001224          2084#   12097
$REG22  001226          2085#   12097
$REG23  001230          2086#   12097
$REG24  001232          2087#   12097
$REG25  001234          2088#   12097
$REG26  001236          2089#    9017    9181
$REG27  001240          2090#
$REG3   001170          2070#   12089
$REG30  001242          2091#
$REG31  001244          2092#
$REG32  001246          2093#
$REG33  001250          2094#
$REG34  001252          2095#
$REG35  001254          2096#
$REG36  001256          2097#    9160*  12092
$REG37  001260          2098#    9161*  12092
$REG4   001172          2071#
$REG5   001174          2072#   6113*   6663*   7534*   8259*   8265*   8273    8279*   8957*   8958*   9157    9525*   9526*
                        9562    9634*   9654*   9666*   9680    9683    12093   12100
$REG6   001176          2073#   6114*   6664*   7535*   8260*   8266*   8275    8280*   9635*   9638*   9649    9651*   9653*
                        9673*   9674*   9675*   9676*   9679*   9681    9682    12093   12100
$REG7   001200          2074#   6115*   6147*   6665*   8267*   8277    8281*   9636*   9637*   9639    9642*   9646*   9648*
                        9680*   9682*   9684*   12093   12100
$RESRE  056170         11624#  11676
$RTNAD  025474          6422#
$R2A  = ****** U        1677
$SAVRE  056132         11608#  11675
$SCOPE  055120          4454   11412#
$SETUP= 000037          4426#   4453    4454    4456    4458    4460    4462    4463    4464    4466    6399    11277   11302
                       11309   11319   11357   11413
$SIZE   055624          6536   11530#
$SIZEX  056066         11568   11578#
$STUP = 177777          4426#
$SUPRS  053702          6570    7235    7420    7480    7507    7972    7985    7998   11105#
$SVLAD  055304         11424   11447#
$SVPC = 000230          1996#   2001
$SWR  = 167000          1775#   1794    1798    1799    1800    1801    1802    1803    1804    1805    2108    2109    2110
                        4463    4464    4466    4467    5347    5371    5397    5445    5484    5532    5588    5624    5648
                        5681    5731    5872    5954    6026    6103    6186    6261    6390    6400    6415    6421    6423
                       11268   11269   11270   11271   11272   11280   11287   11299   11302   11314   11405   11406   11407
                       11408   11409   11415   11427   11429   11430   11431   11438   11439   11440   11450   11453   11456
$SWREG  001342          2131#   4487
$SWRMK= 000000         11409
$TESTN  001324          2122#   5347*   5371*   5397*   5445*   5484*   5532*   5588*   5624*   5648*   5681*   5731*   5872*
                        5954*   6026*   6103*   6186*   6261*   11448*
$TIMES  001304          2108#   4463*   5237*   6400*   7079*   7084*   8444*   11413   11438*   11443   11446*   11456
```

```
$TKB     001146          2058#  6463   6505*  11317  11334  11340
$TKS     001144          2057#  4829*  5019*  5039*  6398*  6474*  6507*  8866*  11317  11332  11338
$TMP0    001262          2099#
$TMP1    001264          2100#  8779*  8780*  8788
$TMP10   001302          2107#  8792*  8793*  8794
$TMP2    001266          2101#  8775*  8803
$TMP3    001270          2102#  8776*  8804
$TMP4    001272          2103#  8777*  8805
$TMP5    001274          2104#
$TMP6    001276          2105#  5689*  5692   5700*  5708*  5710
$TMP7    001300          2106#  5701*  5707*
$TN    = 000022          1784#  1794   5339   5347#  5350   5363   5371#  5374   5387   5397#  5400   5421   5445#
                         5448   5474   5484#  5487   5520   5532#  5535   5572   5588#  5591   5616   5624#  5627
                         5637   5648#  5651   5664   5681#  5684   5723   5731#  5734   5743   5862   5872#  5875
                         5884   5892   5912   5941   5954#  5957   5966   5977   5981   6017   6026#  6029   6038
                         6047   6065   6095   6103#  6106   6177   6186#  6189   6228   6229   6261#  6264
$TNPWR   053616         11053  11054  11074#
$TPB     001152          2060#  10891* 10902
$TPFLG   001157          2064#  10840  10902
$TPS     001150          2059#  10889  10902
$TRAP    056226          4458   11644#
$TRAP2   056250         11655#  11666
$TRP   = 000012         11659#  11668#  11669#  11670#  11671#  11672#  11674   11675#  11676#  11677#  11678#
$TRPAD   056262         11649   11666#
$TSTM    001004          2021#
$TSTNM   001102          2037#   5236*  5321*  6399*  6431*  7081   8450*  9005   11279  11314  11404  11447*  11448
                        11453   11457
$TYPBN=  ****** U       11672
$TYPDS   054170         11207#  11671
$TYPE    052666         10840#  11488  11659  11667
$TYPEC   053100         10870   10877  10884  10889#  10890
$TYPEX   053146         10895   10897  10900#
$TYPOC   053766         11148#  11668
$TYPON   054002         11147   11150#  11670
$TYPOS   053742         11143#  11669
$UNIT    001332          2125#   5285*
$UNITM   001010          2023#
$USWR    001344          2132#
$VECT1   001370          2157#   4557
$VECT2   001372          2158#
$XTSTR   055136         11414   11418#
$$GET4=  000000          6415#
$.$.SM=  ****** U        3378    9759   9994   10185  10196  10234  10548  10589  10601  10655  10677  10695  10722
                        10731
$OFILL   054165         11144*  11148*  11158  11193#
$40CAT=  ****** U       11289   11415
.      = 064424          1975#   1979#  1984#  1986#  1988#  1996   1997#  1999#  2001#  2002#  2008   2009#  2011#
                         2013#   2034#  2114   3488#  3489#  3498#  3520#  3523#  3529#  4418#  4451   4466   4467
                         6132    6208   6212   6311   6423   6427   10902  10974  11032#  11094#  11261#  11314  11317
                        11357   11365  11382#  11456  11457  11509#  12088#
.$ASTA=  ****** U       11461   11464
.$X    = 001000          2008#   2013
..HIGH   000226          1991#
..LOW    000224          1990#
```

```
B4EOPS   6385#  6394
B4SCOP  11399#  11413
CKEXIT   2681#  5788  5896  5916  6051  6069  8860
COMMEN      1#  1932#
ENDCOM      1#  1932#
ERROR    1826#  5766  6156  6165  6673  6969  6990  7536  7786  8261  8262  8264  8282  8295  8544
         8890   8959  9097  9101  9108  9112  9116  9120  9124  9128  9132  9136  9138  9239  9244
         9251   9253  9259  9261  9270  9272  9274  9278  9282  9287  9292  9296  9300  9303  9307
         9311   9315  9319  9323  9327  9332  9337  9342  9347  9352  9357  9367  9380  9382  9392
         9405   9407  9420  9437  9446  9450  9455  9460  9465  9491  9493  9527
ESCAPE      1#  1932#
GETPRI      1#  1932#  11538
GETSWR      1#  1932#
MSG      5339#  5341  5363#  5365  5387#  5389  5421#  5423  5474#  5476  5520#  5522  5572#  5574  5616#
         5618   5637#  5639  5664#  5666  5723#  5725  5862#  5864  5941#  5943  6017#  6019  6095#  6097
         6177#  6179  6227#  6231
MULT        1#  1932#
NEWTST      1#  1932#  5339  5363  5387  5421  5474  5520  5572  5616  5637  5664  5723  5862  5941
         6017   6095  6177  6229
NEXTAD   2680#  5742  5883  5892  5912  5965  5977  5981  6037  6047  6065
NXTAD    2679#  5743  5884  5892  5912  5966  5977  5981  6038  6047  6065
NXTST    2677#  5350  5374  5400  5448  5487  5535  5591  5627  5651  5684  5734  5875  5957  6029
         6106   6189  6264
PARMBK      1#  3262  3294
PARMDF      1#  3168
POP         1#  1932#  10817  11248  11503  11504  11629
PUSH        1#  1932#  10798  11207  11464  11466  11487  11609
P.DRVR      1#  3326
REGDEF      1#  2950
REPORT      1#  1932#
R.DRIV      1#  9688
R.QDRV      1#
SCOPE    1827#  5346  5370  5396  5444  5483  5531  5587  5623  5647  5680  5730  5871  5953  6025
         6102   6185  6260  6394
SETPRI      1#  1932#
SETTRA  11659#  11668  11669  11670  11671  11674  11675  11676  11677
SETUP       1#  1932#  4446
SKIP        1#  1932#
SLASH       1#  1932#
SPACE    1932#
STARS       1#  1776  1782  1932#  1994  2005  2007  2014  2030  2114  2117  4616  4624  4771  4785
         5339   5345  5363  5369  5387  5395  5421  5443  5474  5482  5520  5530  5572  5586  5616
         5622   5637  5646  5664  5679  5723  5729  5851  5853  5862  5870  5941  5952  6017  6024
         6095   6101  6177  6184  6229  6259  6387  6458  6460  6495  6503  6512  6519  6527  6533
         6578   6583  6613  6619  6630  6639  6659  6661  6678  6683  6716  6719  6733  6743  6805
         6815   6879  6887  6906  6911  6926  6941  7044  7048  7061  7092  7113  7190  7198  7215
         7224   7240  7249  7267  7279  7364  7378  7429  7432  7452  7456  7487  7491  7513  7522
         7544   7550  7574  7580  7593  7595  7608  7610  7614  7616  7626  7638  7700  7709  7725
         7727   7747  7756  7790  7792  7802  7805  7822  7825  7843  7847  7866  7869  7892  7900
         7937   7947  7965  7968  7978  7981  7991  7994  8005  8008  8019  8024  8043  8046  8060
         8069   8107  8116  8176  8184  8318  8329  8420  8429  8470  8475  8497  8499  8518  8521
         8574   8579  8623  8626  8648  8652  8670  8675  8706  8709  8734  8738  8764  8774  8807
         8813   8840  8856  8911  8913  8943  8949  8963  8972  9080  9090  9141  9199  9207  9554
         9558   9597  9599  9631  9633  9659  9664  9697  9728  9762  9802  10165  10182  10200  10224
         10271  10284  10303  10329  10348  10370  10428  10455  10733  10746  10789  10825  10905  10915  10933
         10944  10996  11035  11097  11120  11197  11264  11316  11322  11386  11389  11401  11459  11516  11593
```

```
        11638
SWRSU       1#    1775#   1932#   4468#
S.DRVE      1#    3102
TEST     2678#    5339    5363    5387    5421    5474    5520    5572    5616    5637    5664    5723    5862    5941    6017
         6095    6177    6229
TRMTRP  11659#
TYPBIN      1#    1932#
TYPDEC      1#    1932#   6410
TYPNAM      1#    1932#
TYPNUM      1#    1932#
TYPOCS      1#    1932#
TYPOCT      1#    1932#
TYPTXT      1#    1932#
$DECBN      1#
$OCTBN      1#   10731
$$CMRE   2028#    2067    2068    2069    2070    2071    2072    2073    2074    2075    2076    2077    2078    2079    2080
         2081    2082    2083    2084    2085    2086    2087    2088    2089    2090    2091    2092    2093    2094    2095
         2096    2097    2098
$$CMTM   2028#    2099    2100    2101    2102    2103    2104    2105    2106    2107
$$ESCA      1#    1932#
$$NEWT      1#    1932#   5339    5363    5387    5421    5474    5520    5572    5616    5637    5664    5723    5862    5941
         6017    6095    6177    6229
$$SET   11659#   11668   11669   11670   11671   11674   11675   11676   11677
$$SETM   4484#
$$SKIP      1#    1932#
.EQUAT      1#    1775#   1822
.HEADE      1#    1775#   1785
.KT11       1#    1775#   1932
.SETUP      1#    1775#   4426
.SWRHI      1#    1775#   1794
.SWRLO   1775#    1806#   1807    1808    1809    1810    1811    1812    1813
.$ACT1      1#    1775#   1992
.$APTB      1#    2115#
.$APTH      1#    1775#   2003
.$APTY      1#    1775#  11457
.$ASTA      1#
.$CATC      1#    1775#   1973
.$CMTA      1#    1775#   2028
.$DB2D      1#    1775#  11033
.$DB2O      1#    1775#  10994
.$DIV       1#
.$EOP       1#    1775#   6385
.$ERRO      1#    1775#  11262
.$ERRT      1#
.$MULT      1#
.$POWE      1#    1775#
.$RAND      1#    1775#  10787
.$RDDE      1#
.$RDOC      1#
.$READ      1#    1775#  11314
.$R2AZ      1#
.$SAVE      1#    1775#  11591
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    1775#  11399
.$SIZE      1#    1775#  11514
```

```
.$SUPR      1#   1775#  11095
.$TRAP      1#   1775#  11636
.$TYPB      1#
.$TYPD      1#   1775#  11195
.$TYPE      1#   1775#  10823
.$TYPO      1#   1775#  11118
.$40CA      1#
.1170       1#


. ABS.  064424     000


 ERRORS DETECTED:  0

 DSKZ:CZR6ME.BIN,DSKZ:CZR6ME.LST/CRF/SOL/NL:TOC/EQ:QNEWSW=DSKZ:CZR6ME.SML,DRIV10.P11,CZR6ME.P11
 RUN-TIME: 91 132 11 SECONDS
 RUN-TIME RATIO: 494/236=2.0
 CORE USED:  55K  (109 PAGES)
```