RK611

DISKLESS CONT. NO. 5

MD-11-DZR6E-A

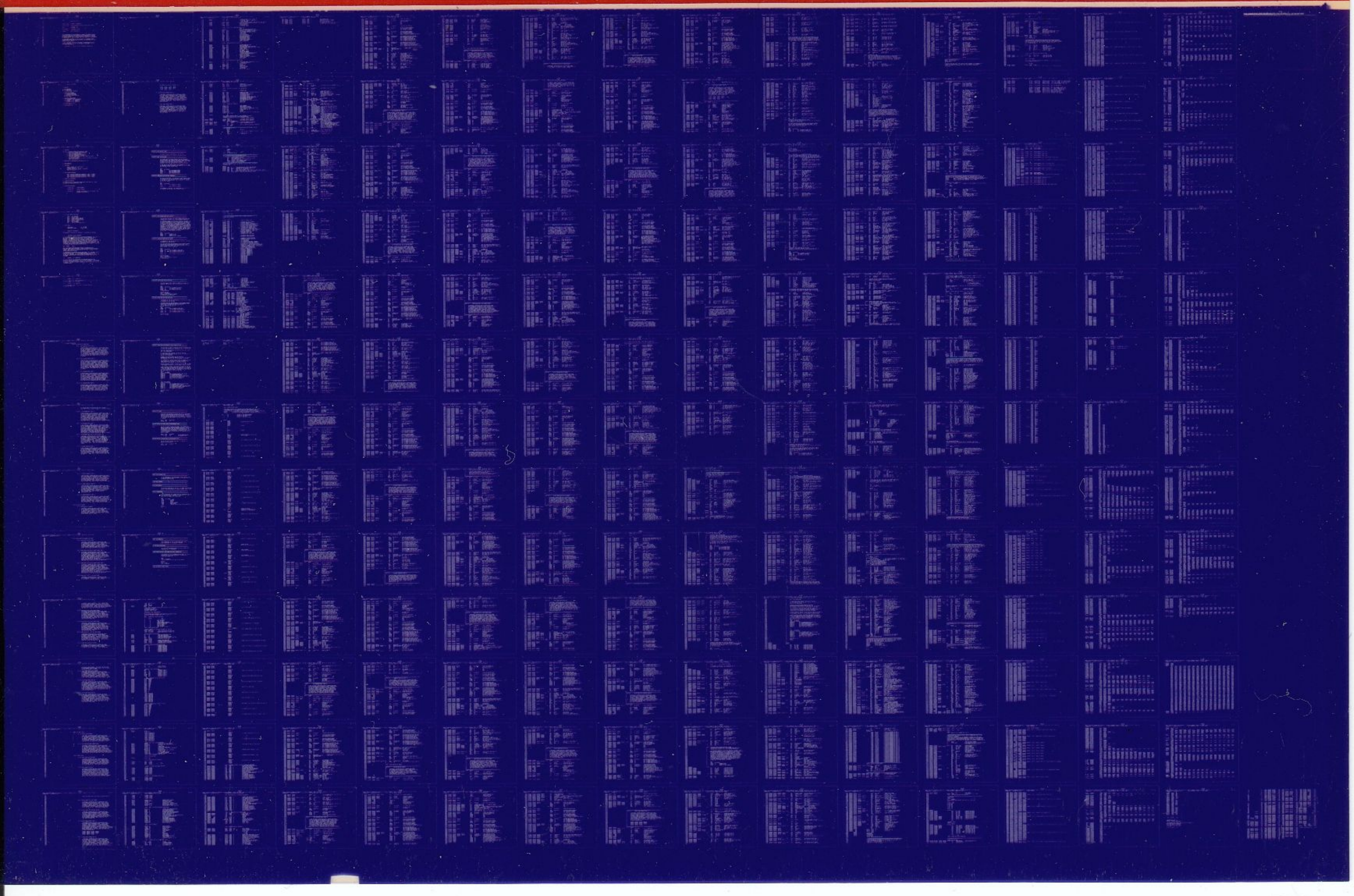
EP-DZR6E-A-DL-A

FICHE 1 OF 1

DEC 1976

digital

MADE IN USA



```
.REM    %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZR6E-A-D

PRODUCT NAME: RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 5

DATE: DECEMBER, 1976

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: MARV TEGROTENHUIS

TABLE OF CONTENTS

1.0 ABSTRACT

THE RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3
- A. TESTS MULTI-SECTOR DATA TRANSFERS.
- B. TESTS MID-TRANSFER SEEKS.
- C. TESTS CYLINDER OVERFLOW CHECKING.
- D. TESTS NPR TRANSFERS TO MEMORY.
- E. TESTS ECC ERROR DETECTION AND CORRECTION, BOTH 16 AND 18 BIT MODE.
- F. TESTS WRITE CHECK BOTH 16 AND 18 BIT MODES AND FORCES WRITE CHECK ERRORS.

NO RK06 DRIVE IS REQUIRED FOR PROGRAM EXECUTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 SYSTEM (16K CORE MEMORY)
CONSOLE TERMINAL
DECTAPE, PAPER TAPE READER, OR DECDISK
RK611 CONTROLLER

2.2 PRELIMINARY PROGRAMS

RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 1 (DZR6A)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (DZR6B)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3 (DZR6C)
RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 4 (DZR6D)

3.0 OPERATING PROCEDURES

3.1 LOADING PROCEDURE

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

3.2 STARTING PROCEDURE

LOCATION 200 - START PROGRAM

LOCATION 204 - RESTART PROGRAM

LOCATION 214 - REQUEST BUS ADDRESS, VECTOR ADDRESS, AND PRIORITY MODIFICATION

3.3 OPTIONAL SWITCH SETTINGS

SW15 - HALT PROGRAM
SW14 - LOOP ON TEST
SW13 - INHIBIT ERROR TYPE OUT
SW12 - ABORT AFTER 20 ERRORS
SW11 - INHIBIT ITERATION COUNT
SW10 - BELL ON ERROR
SW9 - LOOP ON ERROR
SW8 - LOOP ON TEST IN SWITCHES 0-7

3.5 RUN TIME

FIRST PASS	1:40 MINUTES
SUBSEQUENT PASSES	3:40 MINUTES

4.0 OPERATING PROCEDURES

THE PROGRAM IS EXECUTED BY STARTING AT THE APPROPIATE ADDRESS.

4.1 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E., AN 11/04 OR 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RK06 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTINGS ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROES ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

4.2 CONTROL C (↑C) OPERATION

IF ↑C IS TYPED AT ANY TIME DURING THE PROGRAM EXECUTION THE PROGRAM IS HALTED IMMEDIATELY. IF A MONITOR IS PRESENT (XXDP CHAIN, ACT, APT) THE PROGRAM RETURNS CONTROL TO THE MONITOR. IF NO MONITOR IS PRESENT, THE CPU IS HALTED. DEPRESSING THE CONTINUE KEY WILL DO A PROGRAM RESTART.

```
                                4.3     CONTROL S (↑S) OPERATION

                                IF ↑S IS TYPED AT ANY TIME THE PROGRAM WILL GO INTO A STALL LOOP
                                UNTIL A CONTROL Q (↑Q) IS TYPED.

                                4.4     CONTROL Q (↑Q) OPERATION

                                IF A ↑S HAS BEEN TYPED, TYPING THE ↑Q CANCELS THE STALL
                                INITIATED BY THE ↑S.
```

5.0 PROGRAM DESCRIPTION

**MULTI-SECTOR WRITE OPERATIONS

TEST 1 MULTI-SECTOR WRITE (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR. PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION OF HEADER.

TEST 2 MULTI-SECTOR WRITE (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 23. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE TWO ECC WORDS. MAKE SURE THE NEXT HEADER RECOGNITION CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR. PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION OF HEADER.

TEST 3 MID-TRANSFER SEEK (PART 1)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 0, SECTOR 25. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER. CLOCK THROUGH 400 WORDS AND THE TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER. MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 4 MID-TRANSFER SEEK (PART 2)

CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR. PUT THE CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT, CYLINDER 0, HEAD 2, SECTOR 25. CLOCK THROUGH SEEK AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES

```
                                        AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
                                        MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.


TEST 5  MID-TRANSFER SEEK (PART 3)

                                        CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                        DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
                                        CYLINDER 0, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                        AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
                                        TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES
                                        AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
                                        MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.


TEST 6  MID-TRANSFER SEEK (PART 4)

                                        CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                        DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
                                        CYLINDER 0, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK AND
                                        DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                        AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
                                        TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
                                        IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
                                        MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.

TEST 7  MID-TRANSFER SEEK AND 24 SECTOR FORMAT

                                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
                                        DATA OF 401 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                                        CYLINDER 0, HEAD 0, SECTOR 23.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                        AND A GOOD HEADER.  CLOCK THROUGH 400 18 BIT WORDS
                                        AND THE 32 BIT ECC.  CHECK THAT IMPLIED SEEK IS
                                        ISSUE TO NEXT TRACK.  CHECK FOR PROPER HEADER RECOGNITIO
                                        OCCURS.


TEST 10 CYLINDER OVERFLOW (PART 1)

                                        CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                        DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        CYLINDER 632, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                        AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
                                        TWO WORD ECC.  MAKE SURE CYLINDER OVERFLOW ERROR DOES
                                        NOT SET.
```

```
                        TEST 11 CYLINDER OVERFLOW (PART 2)

                                CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                CYLINDER 632, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
                                AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
                                TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.


                        TEST 12 CYLINDER OVERFLOW (PART 3)

                                CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
                                CYLINDER 632, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK
                                AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
                                TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW.
                                ERROR DOES NOT SET.


                        TEST 13 CYLINDER OVERFLOW (PART 4)

                                CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
                                CYLINDER 1456, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK
                                AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
                                TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.


                **NPR WRITING OF MEMORY

                        TEST 14 NPR WRITING OF MEMORY

                                CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                A GOOD HEADER, 40 DATA WORDS.  VERIFY THAT THE WORDS
                                ARE WRITTEN PROPERLY IN MEMORY.
```

```
                        **ECC ERROR DETECTION/CORRECTION TESTS

                TEST 15 READ DATA WITH NO ERROR

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
                        VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.


                TEST 16 READ DATA WITH ONE BIT BURST ERROR (PART 1)

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                        A ONE BIT BURST ERROR OF A 0 THAT SHOULD BE A 1.
                        VERIFY THE COUNTING OF THE POSITION
                        REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                        MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.


                TEST 17 READ DATA WITH ONE BIT BURST ERROR (PART 2)

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                        A ONE BIT BURST ERROR OF A 1 THAT SHOULD BE A 0.
                        VERIFY THE COUNTING OF THE POSITION
                        REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                        MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.


                TEST 20 READ DATA WITH 11 BIT BURST ERROR

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
```

```
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                        AN 11 BIT BURST ERROR.  VERIFY THE COUNING OF THE POSITI
                        REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                        MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.


TEST 21 READ DATA WITH 12 BIT BURST ERROR
                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
                        GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
                        A 12 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
                        POSITION REGISTER.  MAKE SURE DATA CHECK, ECC HARD,
                        AND CONTROLLER ERROR SETS.


TEST 22 READ DATA WITH 23 BIT BURST ERROR

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
                        GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
                        A 23 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
                        POSITION REGISTER.  MAKE SURE DATA CHECK, ECC HARD,
                        AND CONTROLLER ERROR SETS.

TEST 23 READ DATA WITH 1 BIT ECC WORD ERROR

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
                        GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
                        A 1 BIT ERROR IN ECC WORD 2, BIT 6.  VERIFY THE
                        COUNTING OF THE POSITION REGISTER.  MAKE SURE DATA
                        CHECK AND CONTROLLER ERROR SETS.


TEST 24 18 BIT NPR WRITING OF MEMORY

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
```

```
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 40 DATA WORDS.  VERIFY THAT THE WORDS
                        ARE WRITTEN PROPERLY IN MEMORY.

                TEST 25 18 BIT READ DATA WITH NO ERROR
                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
                        VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.


                TEST 26 18 BIT READ DATA WITH DCK

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                        A ONE BIT BURST ERROR BEYOND THE 4128TH BIT OF THE SECTO
                        VERIFY THE COUNTING OF THE POSITION REGISTER AND THE
                        FINAL PATTERN AND POSITION.  MAKE SURE DATA CHECK AND
                        CONTROLLER ERROR SETS.


3491            TEST 27 18 BIT READ DATA WITH ECH

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                        OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 400 DATA WORDS AND AND ECC INDICATING
                        A 12 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
                        POSITION REGISTER.  MAKE SURE DATE CHECK, ECC HARD,
                        AND CONTROLLER ERROR SETS.
```

```
                        **SPECIAL READ TESTS

                                TEST 30 PARTIAL SECTOR READ

                                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                        OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
                                        MAKE SURE ONLY 103 WORDS GOT TRANSFERRED TO MEMORY.


                                TEST 31 SILO UNLOAD BEFORE HEADER ERROR

                                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                        OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
                                        AFTER ECC SIMULATE A SECTOR PULSE AND A BAD SECTOR ERROR
                                        MAKE SURE THE ENTIRE PREVIOUS SECTOR GETS LOADED INTO
                                        MEMORY BEFORE BAD SECTOR ERROR AND CONTROLLER ERROR SETS


                        **WRITE CHECK TESTS

                                TEST 32 WRITE CHECK OF 400 WORDS

                                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                        CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
                                        AND A GOOD 32 BIT ECC.  MAKE SURE NO ERRORS SET.


                                TEST 33 WRITE CHECK WITH DCK

                                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                        CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
                                        AND A 32 BIT ECC INDICATING A 1 BIT BURST ERROR.
                                        VERIFY THAT THE POSITION REGISTER INCREMENTS PROPERLY
                                        AND THAT THE FINAL PATTERN AND POSITION ARE CORRECT.
```

```
                                CHECK THAT DATA CHECK AND CONTROLLER ERROR ARE SET.


TEST 34 WRITE CHECK OF 18 BIT WORDS

        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
        PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A WRITE
        CHECK OF 400 WORDS TC AN RK06 IN 24 SECTOR FORMAT,
        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
        A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
        AND A GOOD 32 BIT ECC.  MAKE SURE NO ERRORS SET.


TEST 35 WRITE CHECK OF A PARTIAL SECTOR

        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
        CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
        CYLINER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
        A GOOD HEADER, TAKE ONE WORD TO BE WRITE-CHECKED, 377 WO
        NOT TO BE WRITE CHECKED, AND A GOOD 32 BIT ECC.
        MAKE SURE NO ERRORS SET.


TEST 36 WRITE CHECK ERROR (PART 1)

        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
        CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
        CLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
        A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 000000
        WRITE CHECKED DATA IS 000001.  MAKE SURE WRITE CHECK ERR
        REPEAT USING EACH OF THE FOLLOWING CONFIGURATIONS
        AS WRITE CHECKED DATA:

        000002  000040  001000  020000
        000004  000100  002000  040000
        000010  000200  004000  100000
        000020  000400  010000


TEST 37 WRITE CHECK ERROR (PART 2)

        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
        CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
        A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 177777
        WRITE CHECKED DATA IS 177776.  MAKE SURE WRITE
        CHECK ERROR SETS.  REPEAT USING EACH OF THE FOLLOWING
```

```
572
```

```
                        CONFIGURATION AS WRITE CHECKED DATA:

                        177775  177737  176777  157777
                        177773  177677  175777  137777
                        177767  177577  173777  077777
                        177757  177377  167777


                TEST 40 WRITE CHECK ERROR (PART 3)

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                        CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, AND ONE WORD OF DATA CONSISTING
                        OF 200000.  WRITE CHECK DATA IS 000000.  MAKE SURE
                        WRITE CHECK ERROR SETS.  REPEAT USING 400000 AS
                        SIMULATED DATA.


                TEST 41 WRITE CHECK ERROR (PART 4)

                        CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                        CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                        CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        AND DRIVE CLEAR MESSAGES SIMULATE A SECTOR PULSE,
                        A GOOD HEADER, 125 GOOD WORDS, AND A BAD WORD.
                        MAKE SURE WRITE CHECK ERROR SET.  CHECK WORD COUNT
                        AND BUS ADDRESS.
```

```
;*****************************************************************
;DIAGNOSTIC MODE SECTOR PULSE
;*****************************************************************

;       THE SECTOR PULSE IS PROVIDED BY THIS ROUTINE

;*****************************************************************
;DIAGNOSTIC MODE IMPLIED SEEK
;*****************************************************************

;       THE CONTROLLER IS CLOCKED THROUGH THE SEEK MESSAGES,
;       GETTING STATUS, AND CLEARING THE DRIVE. SINCE THIS REQUI
;       DIFFERENT COUNT FOR WRITE DATA AND READ DATA/WRITE CHECK
;       THE ROUTINE LOOKS AT THE COMMAND THAT WAS
;       STARTED (L.CS1) AND SETS THE CLOCK COUNT ACCORDINGLY.

;       AT THE END OF THE IMPLIED SEEK THE CONTENTS OF CS1, MR1,
;       AND MR3 ARE CHECKED TO VERIFY THE OPERATION. ALL DISCREP
;       ARE REPORTED IN THE ORDER LISTED.

;       CALL:
;       JSR     R4,DISEEK
;       ERROR   4       ;CS1 MISCOMPARE ERROR
;       ERROR   5       ;MR1 MISCOMPARE ERROR
;       ERROR   6       ;MR2 MISCOMPARE ERROR
;       ERROR   7       ;MR3 MISCOMPARE ERRON

;*****************************************************************
;DIAGNOSTIC MODE READ SYNC PATTERN (PREAMBLE)
;*****************************************************************

;       THE 255 "0" BITS AND 1 "1" BIT ARE CLOCKED THROUTH THE R
;       THE CONTENTS OF MR1 ARE MONITORED TO WATCH FOR AN ERROR.
;       AT THE END OF THE SYNC READ, CS1 IS CHECKED TO INSURE IT
;       DID NOT GET CHANGED.

;       CALL:
;       JSR     R4,DRSYNC
;       ERROR   10      ;MR1 CONTENTS IN ERROR

;       ERROR   11      ;CS1 CONTENTS IN ERROR

;       IF AN ERROR IS DETECTED, THE SUBROUTINE WILL SET UP TO A
;       THE PRESENT TEST UNLESS LOOP ON ERROR IS SET.
```

```
;*********************************************************************
;DIAGNOSTIC MODE HEADER READ AND COMPARE
;*********************************************************************

;       THIS ROUTINE SIMULATES THE THREE HEADER WORDS READ AND
;       COMPARE THAT IS PART OF ALL DATA TRANSFER OPERATIONS.
;
;       THE EXPECTED HEADER IS CALCULATED FROM THE CYLINDER, TRA
;       AND SECTOR SPECIFIED IN THE "L" REGISTERS. THE READING O
;       3 WORDS IS DONE AND MR1 IS MONITORED TO INSURE IT DOES
;       NOT CHANGE. AT THE END OF THE READ/COMPARE, THE CONTENTS
;       OF RKCS1 IS CHECKED AND THE INCREMENT OF RKDCYL AND RKDA
;       VERIFIED. IF THE INCREMENT IS GOOD, THE "L" REGISTERS AR
;       UPDATED TO THE NEW PACK ADDRESS IN CASE OF MULTI-SECTOR
;       OPERATIONS. THE BAD SECTOR ERROR DESIRED (BSEDES) SWITCH
;       IS TESTED AND IF IT IS NON-ZERO, THE 2ND AND 3RD HEADER
;       WORDS ARE CHANGED ACCORDING TO BSEDES CONTENTS. THIS
;       CAN BE USED TO FORCE HEADER TYPE ERPOR.
;
;       CALL:
;       JSR     R4,DHDCMP
;       ERROR   12      ;ERROR IN MR1
;       ERROR   13      ;ERROR IN CS1
;       ERROR   14      ;ERROR IN PACK ADDRESS

;*********************************************************************
;DIAGNOSTIC MODE GAP READ AND SYNC WRITE
;*********************************************************************

;       THE READING OF THE GAP AND WRITING OF THE SYNC (OR PREAM
;       IS HANDLED IN THIS ROUTINE.
;
;       THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
;       ROUTINE AND THE REMAINDER ARE PROCESSED HERE. THEN A
;       CHECK IS MADE THAT WRITE GATE CAME ON AND 255 "0" BITS
;       AND A SINGLE "1" BIT IS WRITTEN.
;
;       CALL:
;       JSR     R4,DWGPSN
;       ERROR   15      ;MR1 IN ERROR IN READ GAP
;       ERROR   16      ;CS1 IN ERROR AFTER READ GAP
;       ERROR   17      ;MR1 IN ERROR IN WRITE SYNC
;
;       ERROR   20      ;CS1 IN ERROR AFTER SYNC WRITE
;
;       ROUTINE CALLED:
;       JSR     PC,WRTBIT
;       RETURN POINT IF ERROR IN WRTBIT
;       NO ERROR RETURN
```

```
                        ;*********************************************************************
                        ;DIAGNOSTIC WRITE DATA AND ECC ROUTINE
                        ;*********************************************************************
                        ;
                        ;       THE DATA IN THE OUPUT BUFFER IS WRITE SIMULATED, ECC IS
                        ;       GENERATED AND WRITTEN, AND THE POSTAMBLE IS WRITTEN.
                        ;
                        ;       CALL:
                        ;       JSR     R4,DWRITE
                        ;       ERROR   17      ;MR1 IN ERROR IN WRITING
                        ;       ERROR   21      ;ECC ERROR IN WRITING
                        ;       ERROR   22      ;CS1 ERROR AFTER WRITE
                        ;       NO ERROR RETURN
                        ;
                        ;       ROUTINE CALLED:
                        ;       JSR     PC,WRTBIT
                        ;       RETURN POINT IF ERROR IN WRTBIT
                        ;       NO ERROR RETURN FROM WRTBIT
                        ;
                        ;*********************************************************************
                        ;DIAGNOSTIC MODE READ GAP AND DATA SYNC
                        ;*********************************************************************
                        ;
                        ;       THE READING OF THE GAP AND READING OF THE SYNC (OR PREAM
                        ;       IS HANDLED IN THIS ROUTINE.
                        ;
                        ;       THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
                        ;       ROUTINE AND THE REMAINING GAP BITS ARE PROCESSED HERE.
                        ;       THEN A CHECK IS MADE THAT READ GATE CAME ON
                        ;       AT THE 129TH BIT OF SYNC.;*
                        ;       CALL:
                        ;       JSR     R4,DRGPSN
                        ;       ERROR   15      ;MR1 IN ERROR IN READ GAP
                        ;       ERROR   16      ;CS1 IN ERROR AFTER READ GAP
                        ;
                        ;       ERROR   32      ;MR1 IN ERROR IN READ SYNC
                        ;       ERROR   33      ;CS1 IN ERROR AFTER SYNC READ
                        ;
                        ;       ROUTINE CALLED:
                        ;       JSR     PC,RDBIT
```

```
;*****************************************************************
;DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE
;*****************************************************************

;       THIS ROUTINE IS USED TO SIMULATE THE READ OR WRITE CHECK
;       OF DATA. THE ROUTINE WILL TRANSFER A FULL OR PARTIAL SEC
;
;       THE FIRST WORD AFTER THE CALL SPECIFIES HOW MANY WORDS
;       ARE TO BE TRANSFERRED.
;
;       IF LESS THAN A FULL SECTOR, THE ECC AT THE END
;       OF THE TRANSFER IS NOT CHECKED. THE ECC IS CHECKED AT TH
;       EACH BIT IS TRANSFERED.
;
;       EITHER 16 OR 18 BIT WORDS CAN BE SIMULATED
;       DEPENDING ON THE STATE OF CFMT BIT IN L.CS1. IF 18 BITS
;       ARE REQUIRED, BITS 16 AND 17 ARE ASSUMED TO BE ZERO.
;
;       THE CONTENTS OF LOCATION ECPOSX MUST BE LOADED WITH THE
;       COUNT EXPECTED AT THE END OF THE DATA TRANSFER. THIS
;       LOADING MUST OCCUR BEFORE THE ROUTINE IS CALLED.
;
;       ANOTHER LOCATION, ECPATX, IS SET TO 0 IF THE ECC IS EXPE
;       TO BE CORRECT AND SET TO 1 IF AN ERROR IS BEING FORCED.
;
;       A 3RD LOCATION, ECCSRC, SPECIFIES THE SOURCE OF THE ECC
;       WORDS TO BE USED IN CHECKING THE OPERATION. IF ECCSRC IS
;       1, THE ECC WORDS FED INTO THE DECODER ARE TAKEN FROM THE
;       LAST TWO LOCATIONS OF OBUFF, IF ECCSRC IS 0, THE COMPUTE
;       ECC WORDS FROM ECCHI AND ECCLO ARE USED.
;       CALL:
;       JSR     R4,DREAD
;       LENGTH OF TRANSFER
;       ERROR 34        ;MR1 IN ERROR READING DATA OR ECC
;       ERROR 35        ;ECC ERROR READING DATA
;       ERROR 36        ;CS1 ERROR AFTER READING DATA OR ECC
;       ERROR 41        ;ECC REGISTER INCORRECT AFTER READ ECC
;       ERROR 42        ;ERR IN ECC PAT CALCULATION
;       ERROR 43        ;ERR IN ECC POS COUNTING
;
;       OR
;
;       JSR     R4,DWRTCK
;       LENGTH OF TRANSFER
;       ERROR 53        ;MR1 IN ERROR WRITE CHK OR ECC READ
;       ERROR 54        ;ECC ERROR IN WRITE CHECK
;       ERROR 55        ;CS1 ERROR AFTER IN WRT CHK OR ECC READ
;       ERROR 41        ;ECC REGISTER INCORRECT AFTER READ ECC
;       ERROR 42        ;ERR IN ECC PAT CALCULATION
;       ERROR 43        ;ERR IN ECC POS COUNTING
;
;       ROUTINES CALLED:
;       RDBIT
;       ECCGEN
```

```
                        ;*****************************************************************
                        ;GENERATE ECC WORD
                        ;*****************************************************************
                        ;
                        ;       EACH TIME THIS ROUTINE IS CALLED THE ECC IS CALCULATED F
                        ;       NEXT BIT TO BE READ OR WRITTEN AND THE BITS PREVIOUSLY
                        ;       READ OR WRITTEN (STORED IN ECCHI AND ECCLO). THE RESULTS
                        ;       THE CALCULATION ARE PLACED IN ECCHI AND ECCLO. THE LOW 0
                        ;       11 BITS OF ECCHI ARE PLACED IN E.ECPT FOR EASY COMPARISO
                        ;       TO RKECPT.
                        ;
                        ;       CALL:   JSR     PC,ECCGEN
                        ;       RETURN: RTS     R4
                        ;
                        ;*****************************************************************
                        ;SIMULATE ONE BIT OF WRITE DATA IN MAINTANENCE MODE
                        ;*****************************************************************
                        ;
005835                  ;       ONE BIT OF DATA IS WRITE SIMULATED. THE CONTENTS OF
                        ;       RKMR1 IS COMPUTED (PRECOMP ADVANCE AND DELAY, MAINTENANC
                        ;       ENCODED WRITE DATA) FOR THE PROPER TRANSITION OF THE
                        ;       MAINTENANCE CLOCK AND CHECKED AT EACH TRANSITION. IF MR1
                        ;       IS INCORRECT AT ANY TRANSITION, AN ERROR IS SET UP TO
                        ;       CONTAIN A MESSAGE THAT IDENTIFIES THE TRANSITION WHEN TH
                        ;       ERROR OCCURRED.
                        ;
                        ;       CALL:   JSR     PC,WRTBIT
                        ;       RETURN: RTS     PC+2            FOR NO ERROR RETURN
                        ;               RTS     PC              FOR ERROR IN MR1
                        ;
                        ;*****************************************************************
                        ;SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE
                        ;*****************************************************************
                        ;
                        ;       ONE BIT OF DATA IS READ SIMULATED. NO CHECKING IS
                        ;       DONE BY THIS ROUTINE.
                        ;
                        ;       CALL:   JSR     PC,RDBIT
                        ;       RETURN: RTS     PC
```

```
                        ;*********************************************************************
                        ;MEMORY CHECK ENABLE TRAP
                        ;*********************************************************************
                        ;
                        ;       IF THE PROCESSOR IN USE HAS MEMORY PARITY OPTION,
                        ;       IT WILL BE ENABLED. THIS ROUTINE WILL PROCESS TRAPS CAUS
                        ;       BY MEMORY PARITY ERRORS.
                        ;
                        ;*********************************************************************
                        ;CLEAR INPUT BUFFER
                        ;*********************************************************************
                        ;
                        ;*********************************************************************
                        ;BUILD DATA BUFFER
                        ;*********************************************************************
                        ;
                        ;       THE PATTERN SPECIFIED IN THE CALL IS LOADED INTO THE DAT
                        ;       BUFFER. THE ENTIRE BUFFER IS ALWAYS LOADED (400(8) WORDS
                        ;
                        ;*********************************************************************
                        ;LOAD "L" REGISTERS
                        ;*********************************************************************
                        ;
                        ;THE "L" REGISTERS ARE LOADED FORM THE PARAMETERS FOLLOWI
                        ;THE SUBROUTINE CALL. DIAGNOSTIC MODE IS SET IN L.MR1
                        ;AND L.CS1 IS LOADED WITH THE COMMAND.
                        ;
                        ;CALL:
                        ;JSR     R4,LOADRK
                        ;.WORD           CYLINDER
                        ;.BYTE           ;SECTOR
                        ;.BYTE           ;TRACK
                        ;.WORD           ;BUFFER ADDRESS
                        ;.WORD           ;WORD COUNT
                        ;.WORD           ;COMMAND,FORMAT,DRIVE TYPE,&UPPER BUS AD
                        ;
                        ;RETURN:
                        ;RTS     R4
```

```
;*********************************************************************
;START THE OPERATION
;*********************************************************************

;               THE INFORMATION IN THE "L" REGISTERS ARE LOADED INTO THE
;               RK611 REGISTERS IN A STARAIGHT TRANSFER.

;*********************************************************************
;GET THE RK611 REGISTERS
;*********************************************************************

;               ALL THE RK611 REGISTERS EXCEPT THE DATA BUFFER ARE
;               STORED IN THE "T" REGISTERS.

;*********************************************************************
;"FIRST SHALL BE LAST, LAST SHALL BE FIRST" SUBROUTINE
;*********************************************************************

;               THE CONTENTS OF R3 IS SWAPPED END FOR END, I.E. BIT 15
;               BECOMES BIT 0 AND VICE VERSA,BIT 14 BECOMES BIT
;               1 AND VICE VERSA, ETC.

;               CALL:
;               JSR     R4,FSBLVV
;               WITH R3 LOADED WITH THE WORD TO BE SWAPPED

;               RETURN:
;               RTS     R4
;               WITH R3 SWAPPED.

;*********************************************************************
;CHECK FOR MEMORY CHECK ENABLE
;*********************************************************************
```

```
                                %
                        .NLIST  CND,MD,MC,TOC
                        .LIST   ME
                        .ENABL  ABS,AMA
167400                  $SWR=   167400
000001                  $TN=    1
                        .TITLE  RK611 DISKLESS CONTROLLER DIAGNOSTIC: P5 MD-11-DZR6E-A
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY MARV TEGROTENHUIS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            ABORT PROGRAM AFTER 20 ERRORS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>
                        .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
```

```
                        SP=     %6              ;;STACK POINTER
                        PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
```

```
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

000250                  MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

177572                  SR0=    177572
177574                  SR1=    177574
177576                  SR2=    177576
172516                  SR3=    172516

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

172300                  KIPDR0= 172300
172302                  KIPDR1= 172302
172304                  KIPDR2= 172304
172306                  KIPDR3= 172306
172310                  KIPDR4= 172310
172312                  KIPDR5= 172312
172314                  KIPDR6= 172314
172316                  KIPDR7= 172316

                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

172340                  KIPAR0= 172340
172342                  KIPAR1= 172342
172344                  KIPAR2= 172344
172346                  KIPAR3= 172346
```

```
1087            172350          KIPAR4= 172350
1088            172352          KIPAR5= 172352
1089            172354          KIPAR6= 172354
1090            172356          KIPAR7= 172356
1091
1092            000114          MEMVEC= 114                     ;MEMORY PARITY VECTOR
1093            172100          MEMBAS= 172100                  ;MEM PARITY OPTION
1094            000004          WR.PAR= 4                       ;WRITE BAD PARITY
1095            000001          PAR.EN= 1                       ;ENABLE PARITY ENABLE
1096            120210          AVECT1= 120210                  ;DEFINE RK611 VECTOR ADDRESS
1097            000005          APRIOR= 5                       ;DEFINE RK611 PRIORITY
1098            177440          ABASE=  177440                  ;DEFINE BASE OF RK611 REGISTERS
1099
1100                            .SBTTL  RK611 CONTROLLER REGISTER DEFINITION
1101
1102            000000          RKCS1=  0                       ;CONTROL AND STATUS REGISTER 1
1103            000002          RKWC=   2                       ;WORD COUNT REGISTER
1104            000004          RKBA=   4                       ;BUS ADDRESS REGISTER
1105            000006          RKDA=   6                       ;DESIRED TRACK SECTOR REGISTER
1106            000010          RKCS2=  10                      ;CONTROL AND STATUS REGISTER 2
1107            000012          RKDS=   12                      ;DRIVE STATUS REGISTER
1108            000014          RKER=   14                      ;ERROR REGISTER
1109            000016          RKASOF= 16                      ;ATTENTION SUMMARY AND OFFSET REGISTER
1110            000020          RKDCYL=20                       ;DESIRED CYLINDER REGISTER
1111            000024          RKDB=   24                      ;DATA BUFFER
1112            000026          RKMR1=  26                      ;MAINTENANCE REGISTER 1
1113            000034          RKMR2=  34                      ;MAINTENANCE REGISTER 2
1114            000036          RKMR3=  36                      ;MAINTENANCE REGISTER 3
1115            000030          RKECPS= 30                      ;ECC POSITION INFORMATION
1116            000032          RKECPT= 32                      ;ECC PATTERN INFORMATION
1117            000022          RKSPAR= 22                      ;SPARE REGISTER
1118
1119                            .SBTTL  DRIVE COMMANDS
1120
1121            000001          SELDRV= 01                      ;SELECT DRIVE
1122            000003          PACK=   03                      ;PACK ACKNOWLEDGE
1123            000005          CLEAR=  05                      ;DRIVE CLEAR
1124            000007          UNLOAD= 07                      ;UNLOAD
1125            000011          SRTSPL= 11                      ;START SPINDLE
1126            000013          RECAL=  13                      ;RECALIBRATE
1127            000015          OFFSET= 15                      ;OFFSET
1128            000017          SEEK=   17                      ;SEEK
1129            000021          RDDATA= 21                      ;READ DATA
1130            000023          WRDATA= 23                      ;WRITE DATA
1131            000025          RDHEAD= 25                      ;READ HEADER
1132            000027          WRHEAD= 27                      ;WRITE HEADER AND DATA
1133            000031          WRTCHK= 31                      ;WRITE CHECK
1134            000300          INTR=   300                     ;GENERATE INTERRUPT TO CPU
1135
1136                            .SBTTL  CONTROL AND STATUS REGISTER 1 BITS
1137
1138            000001          GO=     BIT0                    ;GO BIT
1139            000100          IE=     BIT6                    ;INTERRUPT ENABLE
1140            000200          RDY=    BIT7                    ;CONTROLLER READY
1141            000400          BA16=   BIT8                    ;BUS ADDRESS BIT 16
1142            001000          BA17=   BIT9                    ;BUS ADDRESS BIT 17
```

```
1143            002000          CDT=    BIT10                   ;CONTROLLER DRIVE TYPE (0=RK06)
1144            004000          CTO=    BIT11                   ;CONTROLLER TIMED OUT WAITING FOR
1145                                                            ;   DRIVE RESPONSE
1146            010000          CFMT=   BIT12                   ;CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
1147            020000          SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
1148            040000          DI=     BIT14                   ;DRIVE INTERRUPT
1149            100000          CERR=   BIT15                   ;CONTROLLER ERROR
1150            100000          CCLR=   BIT15                   ;CONTROLLER CLEAR
1151
1152                            .SBTTL  CONTROL AND STATUS REGISTER 2 BITS
1153
1154            000007          DRVMSK= 7                       ;MASK FOR DRIVE SELECTION CODE
1155            000010          RLS=    BIT3                    ;DESELECT OR RELEASE DRIVE IN BITS 0-2
1156            000020          BAI=    BIT4                    ;BUS ADDRESS INCREMENT INHIBIT
1157            000040          SCLR=   BIT5                    ;CLEAR CONTROLLER AND ALL DRIVES
1158            000100          IR=     BIT6                    ;INPUT READY
1159            000200          OR=     BIT7                    ;OUTPUT READY
1160            000400          UFE=    BIT8                    ;UNIT FIELD ERROR
1161            001000          MDS=    BIT9                    ;MULTIPLE DRIVE SELECT
1162            002000          PGE=    BIT10                   ;PROGRAMMING ERROR
1163            004000          NEM=    BIT11                   ;NON-EXISTENT MEMORY
1164            010000          NED=    BIT12                   ;NON-EXISTENT DRIVE
1165            020000          UPE=    BIT13                   ;UNIBUS PARITY ERROR
1166            040000          WCE=    BIT14                   ;WRITE CHECK ERROR
1167            100000          DLT=    BIT15                   ;DATA LATE ERROR
1168
1169                            .SBTTL  ERROR REGISTER BIT DEFINITION
1170
1171            000001          ILF=    BIT0                    ;ILLEGAL FUNCTION CODE
1172            000002          SKI=    BIT1                    ;SEEK INCOMPLETE
1173            000004          NXF=    BIT2                    ;NON-EXECUTABLE DRIVE FUNCTION
1174            000010          DRPAR=  BIT3                    ;DRIVE DETECTED DRIVE BUS PARITY ERROR
1175            000020          FMTE=   BIT4                    ;FORMAT ERROR
1176            000040          DTYE=   BIT5                    ;DRIVE TYPE ERROR
1177            000100          ECH=    BIT6                    ;ECC HARD
1178            000200          BSE=    BIT7                    ;BAD SECTOR ERROR
1179            000400          HVRC=   BIT8                    ;HEADER VRC ERRROR
1180            001000          COE=    BIT9                    ;CYLINDER ADDRESS OVERFLOW ERROR
1181            002000          IDAE=   BIT10                   ;INVALID DISK ADDRESS ERROR
1182            004000          WLE=    BIT11                   ;WRITE LOCK ERROR
1183            010000          DTE=    BIT12                   ;DRIVE TIMING ERROR
1184            020000          OPI=    BIT13                   ;OPERATION (SEARCH) INCOMPLETE
1185            040000          UNS=    BIT14                   ;DRIVE UNSAFE
1186            100000          DCK=    BIT15                   ;DATA CHECK
1187
1188                            .SBTTL  STATUS REGISTER BIT DEFINITION
1189
1190            000001          DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
1191                                                            ;   THIS BIT IS RESET)
1192            000004          OFST=   BIT2                    ;DRIVE OFFSET
1193            000010          ACLO=   BIT3                    ;AC LOW
1194            000020          SPDLSS= BIT4                    ;SPEED LOSS
1195            000040          DROT=   BIT5                    ;DRIVE OFF TRACK
1196            000100          VV=     BIT6                    ;VOLUME VALID
1197            000200          DRDY=   BIT7                    ;DRIVE READY
1198            000400          DDT=    BIT8                    ;DRIVE TYPE (0=RK06)
```

```
1199            004000                  WRL=    BIT11                   ;WRITE LOCK
1200            020000                  PIP=    BIT13                   ;POSITIONING IN PROGRESS
1201            040000                  DSC=    BIT14                   ;DRIVE STATUS CHANGE
1202            100000                  SVAL=   BIT15                   ;STATUS VALID
1203
1204                                    .SBTTL  MAINTENANCE REGISTER 1 BIT DEFINITION
1205
1206            000017                  MESMSK= 17                      ;MESSAGE MASK
1207
1208            000020                  PAT=    BIT4                    ;FORCE EVEN PARITY ON DRIVE MESSAGE LINES
1209            000040                  DMD=    BIT5                    ;DIAGNOSTIC MODE
1210            000100                  MSP=    BIT6                    ;MAINTENANCE SECTOR PULSE
1211            000200                  MIND=   BIT7                    ;MAINTENANCE INDEX
1212            000400                  MCLK=   BIT8                    ;MAINTENANCE CLOCK
1213            001000                  MERD=   BIT9                    ;MAINTENANCE ENCODED READ DATA
1214            002000                  MEWD=   BIT10                   ;MAINTENACNE ENCODED WRITE DATA
1215            004000                  PCA=    BIT11                   ;PRECOMPENSATION ADVANCE
1216            010000                  PCD=    BIT12                   ;PRECOMPENSATION DELAY
1217            020000                  ECCW=   BIT13                   ;ECC WORD IS BEING READ OR WRITTEN
1218            040000                  WRTGAT= BIT14                   ;WRITE GATE
1219            100000                  RDGATE= BIT15                   ;READ GATE
1220
1221                                    .SBTTL  TRANSMITTED MESSAGE A
1222
1223            000020                  S.SEEK= BIT4                    ;SEEK COMMAND
1224            000040                  S.RECL= BIT5                    ;RECALIBRATE COMMAND
1225            000100                  S.STSP= BIT6                    ;START SPINDLE COMMAND
1226            000200                  S.RTC=  BIT7                    ;DRIVE RETURN TO CENTERLINE COMMAND
1227            000400                  S.CLR=  BIT8                    ;CLEAR ERROR AND DSC
1228            001000                  S.FMT=  BIT9                    ;FORMAT
1229            002000                  S.UNLD= BIT10                   ;UNLOAD
1230            004000                  S.PACK= BIT11                   ;SET VOLUME VALID (PACK ACKNOWNLEDGE)
1231                                    .SBTTL  TRAP CATCHER
1232
1233            000000                          .=0
1234                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1235                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1236                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1237            000174                          .=174
1238  000174   000000                  DISPREG: .WORD  0                       ;;SOFTWARE DISPLAY REGISTER
1239  000176   000000                  SWREG:   .WORD  0                       ;;SOFTWARE SWITCH REGISTER
1240                                    .SBTTL  STARTING ADDRESS(ES)
1241  000200   000137  002400                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1242  000204   000137  002370                  JMP     RESTRT          ;JUMP TO RESTART ROUTINE
1243            000214                        .=214
1244  000214   000137  002360                  JMP     PARM            ;JUMP TO OPERATOR ASSIGNED PARMETERS
1245                                    .SBTTL  ACT11 HOOKS
1246
1247                                    ;;*****************************************************************
1248                                    ;HOOKS REQUIRED BY ACT11
1249            000220                          $SVPC=.                 ;SAVE PC
1250            000046                          .=46
1251  000046   025042                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1252            000052                          .=52
1253  000052   000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
1254            000220                          .=$SVPC                 ;; RESTORE PC
```

```
                    000114                          .=MEMVEC
                    033524                          MEMERR
                    000340                          PR7
                    001000                          .=1000
                                        .SBTTL  APT PARAMETER BLOCK


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

                    001000              $APTHD:
                    000000              $HIBTS: .WORD   0           ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
                    001250              $MBADR: .WORD   $MAIL       ;;ADDRESS OF APT MAILBOX (BITS 0-15)
                    000132              $TSTM:  .WORD   90.         ;;RUN TIM OF LONGEST TEST
                    001604              $PASTM: .WORD   900.        ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
                    000144              $UNITM: .WORD   100.        ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
                    000030                      .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

                                ;;**********************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560          ;;TTY KBD STATUS
001146  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $REGAD: .WORD   0       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000                  $REG0:  .WORD   0       ;;CONTAINS (($REGAD)+0)
001164  000000                  $REG1:  .WORD   0       ;;CONTAINS (($REGAD)+2)
001166  000000                  $REG2:  .WORD   0       ;;CONTAINS (($REGAD)+4)
001170  000000                  $REG3:  .WORD   0       ;;CONTAINS (($REGAD)+6)
001172  000000                  $REG4:  .WORD   0       ;;CONTAINS (($REGAD)+10)
001174  000000                  $REG5:  .WORD   0       ;;CONTAINS (($REGAD)+12)
001176  000000                  $REG6:  .WORD   0       ;;CONTAINS (($REGAD)+14)
001200  000000                  $REG7:  .WORD   0       ;;CONTAINS (($REGAD)+16)
001202  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001204  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001206  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001210  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
001212  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
001214  000000                  $TMP5:  .WORD   0       ;;USER DEFINED
001216  000000                  $TMP6:  .WORD   0       ;;USER DEFINED
001220  000000                  $TMP7:  .WORD   0       ;;USER DEFINED
001222  000000                  $TMP10: .WORD   0       ;;USER DEFINED
```

```
1337  001224  000000                $TMP11: .WORD   0               ;;USER DEFINED
1338  001226  000000                $TMP12: .WORD   0               ;;USER DEFINED
1339  001230  000000                $TMP13: .WORD   0               ;;USER DEFINED
1340  001232  000000                $TMP14: .WORD   0               ;;USER DEFINED
1341  001234  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
1342  001236  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
1343  001240  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
1344  001244     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
1345  001245     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
1346  001246  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
1347                                ;;**************************************************************
1348                                .SBTTL  APT MAILBOX-ETABLE
1349
1350                                ;;**************************************************************
1351                                .EVEN
1352  001250                        $MAIL:                          ;;APT MAILBOX
1353  001250  000000                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
1354  001252  000000                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
1355  001254  000000                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
1356  001256  000000                $PASS:  .WORD   APASS           ;;PASS COUNT
1357  001260  000000                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
1358  001262  000000                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
1359  001264  000000                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
1360  001266  000000                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
1361  001270                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
1362  001270     000                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
1363  001271     000                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
1364  001272  000000                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
1365  001274  000000                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
1366  001276  000000                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
1367                                ;*                              BITS 15-11=CPU TYPE
1368                                ;*                                   11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1369                                ;*                                   11/70=06,PDQ=07,Q=10
1370                                ;*                              BIT 10=REAL TIME CLOCK
1371                                ;*                              BIT  9=FLOATING POINT PROCESSOR
1372                                ;*                              BIT  8=MEMORY MANAGEMENT
1373  001300     000                $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
1374  001301     000                $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
1375                                ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
1376                                ;*                                   900 NSEC CORE=001
1377                                ;*                                   300 NSEC BIPOLAR=002
1378                                ;*                                   500 NSEC MOS=003
1379  001302  000000                $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
1380                                ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
1381  001304     000                $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
1382  001305     000                $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
1383  001306  000000                $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
1384  001310     000                $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
1385  001311     000                $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
1386  001312  000000                $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
1387  001314     000                $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
1388  001315     000                $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
1389  001316  000000                $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
1390  001320  120210                $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1391  001322  000000                $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1392  001324  177440                $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
```

```
  1393  001326  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
  1394  001330                        $ETEND:
  1395                                .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

001330                          $ERRTB:
                                ;       ERROR 1
001330  044512                          EM000                   ;UNEXPECTED PARITY ERROR
001332  043071                          DH000C
001334  041316                          DT000
001336  041634                          DF000
                                ;       ERROR 2                 ;UNUSED
001340  000000                          0
001342  000000                          0
001344  000000                          0
001346  000000                          0
                                ;       ERROR 3                 ;UNUSED
001350  000000                          0
001352  000000                          0
001354  000000                          0
001356  000000                          0
                                ;       ERROR 4                 ;ERROR ATTEMPTING IMPLIED SEEK
001360  044557                          EM5004                  ;CS1 IN ERROR
001362  044623                          E5004A
001364  041322                          DT0004
001366  041640                          DF0004
                                ;       ERROR 5                 ;MR1 IN ERROR ATTEMPTING IMPLIED SEEK
001370  044557                          EM5004
001372  044642                          E5004B
001374  041340                          DT0005
001376  041670                          DF0005
                                ;       ERROR 6                 ;MR2 IN ERROR ATTEMPTING IMPLIED SEEK
001400  044557                          EM5004
001402  044661                          E5004C
001404  041350                          DT0006
001406  041714                          DF0006
                                ;       ERROR 7                 ;MR3 IN ERROR ATTEMPTING IMPLIED SEEK
001410  044557                          EM5004
001412  044700                          E5004D
001414  041360                          DT0007
001416  041740                          DF0007
                                ;       ERROR 10                ;MR1 ERROR READING PREAMBLE
001420  044771                          EM5010
```

```
1452  001422  044642                  E5004B
1453  001424  041370                  DT0010
1454  001426  041764                  DF0010
1455
1456                          ;       ERROR 11            ;CS1 ERROR READING PREAMBLE
1457  001430  044771                  EM5010
1458  001432  044623                  E5004A
1459  001434  041406                  DT0011
1460  001436  042010                  DF0011
1461
1462                          ;       ERROR 12            ;MR1 IN ERROR HEADER READ/COMPARE
1463  001440  045051                  EM5011
1464  001442  044642                  E5004B
1465  001444  041370                  DT0010
1466  001446  041764                  DF0010
1467
1468                          ;       ERROR 13            ;CS1 IN ERROR HEADER READ/COMPARE
1469  001450  045051                  EM5011
1470  001452  044623                  E5004A
1471  001454  041406                  DT0011
1472  001456  042010                  DF0011
1473
1474                          ;       ERROR 14            ;PACK ADDRESS ERROR HDR READ/COMPARE
1475  001460  045051                  EM5011
1476  001462  045124                  E5011A
1477  001464  041432                  DT0014
1478  001466  042040                  DF0014
1479
1480                          ;       ERROR 15            ;MR1 ERROR READING GAP
1481  001470  045154                  EM5015
1482  001472  044642                  E5004B
1483  001474  041370                  DT0010
1484  001476  041764                  DF0010
1485
1486                          ;       ERROR 16            ;CS1 ERROR AFTER READING GAP
1487  001500  044771                  EM5010
1488  001502  044623                  E5004A
1489  001504  041406                  DT0011
1490  001506  042010                  DF0011
1491
1492                          ;       ERROR 17            ;ERROR IN WRITING
1493  001510  000000          EMW:    0                   ;REGISTER IN ERROR AND CLOCK
1494  001512  000000                  0                   ;TRANSITION IS IDENTIFIED
1495  001514  041446                  DT0017
1496  001516  042064                  DF0017
1497
1498                          ;       ERROR 20            ;ERROR IN CS1 AFTER WRITING SYNC
1499  001520  045211                  EM5020
1500  001522  044623                  E5004A
1501  001524  041406                  DT0011
1502  001526  042010                  DF0011
1503
1504                          ;       ERROR 21            ;ERROR IN ECC WORDS
1505  001530  045270                  EM5021
1506  001532  045336                  E5021A
1507  001534  041470                  DT0021
```

```
1508  001536  042110                          DF0021
1509
1510                                  ;       ERROR 22                ;ERR IN CS1 AFTER WRITE DATA
1511  001540  045270                          EM5021
1512  001542  044623                          E5004A
1513  001544  041406                          DT0011
1514  001546  042010                          DF0011
1515
1516                                  ;       ERROR 23                ;ERROR IN MR2
1517  001550  045366                          EM5023                  ;DOING MULTI-SECTOR WRITE
1518  001552  044661                          E5004C
1519  001554  041350                          DT0006
1520  001556  041714                          DF0006
1521
1522                                  ;       ERROR 24                ;MR3 IN ERROR
1523  001560  045366                          EM5023                  ;DOING MULTI-SECTOR WRITE
1524  001562  044700                          E5004D
1525  001564  041360                          DT0007
1526  001566  041740                          DF0007
1527
1528                                  ;       ERROR 25                ;CS1 IN ERROR
1529  001570  045366                          EM5023                  ;AFTER MULTI-SECTOR WRITE
1530  001572  044623                          E5004A
1531  001574  041322                          DT0004
1532  001576  041640                          DF0004
1533
1534                                  ;       ERROR 26
1535  001600  046006                          EM5026                  ;NO COE ERROR
1536  001602  000000                  EH026:  .WORD   0
1537  001604  041502                          DT0026
1538  001606  042134                          DF0026
1539
1540                                  ;       ERROR 27
1541  001610  046146                          EM5027                  ;OTHER ERROR WHILE FORCING COE
1542  001612  043040                          DH000A
1543  001614  041502                          DT0026
1544  001616  042134                          DF0026
1545
1546                                  ;       ERROR 30
1547  001620  046320                          EM5030                  ;UNEXPECTED CYL OVERFLOW
1548  001622  000000                  EH030:  .WORD   0
1549  001624  041502                          DT0026
1550  001626  042134                          DF0026
1551
1552                                  ;       ERROR 31
1553  001630  046413                          EM5031                  ;UNEXPECTED ERROR TESTING COE
1554  001632  000000                  EH031:  .WORD   0
1555  001634  041502                          DT0026
1556  001636  042134                          DF0026
1557
1558                                  ;       ERROR 32                ;MR1 ERROR READING DATA SYNC
1559  001640  046500                          EM5032
1560  001642  044642                          E5004B
1561  001644  041370                          DT0010
1562  001646  041764                          DF0010
1563
```

```
1564                                    ;       ERROR 33                ;CS1 ERROR AFTER READING SYNC
1565  001650  046500                            EM5032
1566  001652  044623                            E5004A
1567  001654  041406                            DT0011
1568  001656  042010                            DF0011
1569
1570                                    ;       ERROR 34                ;MR1 IN ERROR READING DATA OR ECC
1571  001660  046556                            EM5034
1572  001662  044642                            E5004B
1573  001664  041370                            DT0010
1574  001666  041764                            DF0010
1575
1576                                    ;       ERROR 35                ;ECC ERROR READING DATA
1577  001670  046623                            EM5035
1578  001672  045336                            E5021A
1579  001674  041470                            DT0021
1580  001676  042110                            DF0021
1581
1582                                    ;       ERROR 36                ;CS1 ERROR AFTER READING DATA OR ECC
1583  001700  046556                            EM5034
1584  001702  044623                            E5004A
1585  001704  041406                            DT0011
1586  001706  042010                            DF0011
1587
1588                                    ;       ERROR 37                ;DATA COMPARE ERROR (1ST MESSAGE)
1589  001710  046664                            EM5037
1590  001712  043040                            DH000A
1591  001714  041524                            DT0037
1592  001716  042170                            DF0037
1593
1594                                    ;       ERROR 40                ;DATA COMPARE ERROR (2ND THRU 10TH ERR)
1595  001720  000000                            0
1596  001722  000000                            0
1597  001724  041530                            DT0040
1598  001726  042210                            DF0040
1599
1600                                    ;       ERROR 41
1601  001730  046740                            EM5041                  ;ECC REGISTER INCORRECT AFTER ECC READ
1602  001732  043040                            DH000A
1603  001734  041536                            DT0041
1604  001736  042214                            DF0041
1605
1606                                    ;       ERROR 42
1607  001740  047025                            EM5042                  ;ERROR IN ECC PAT CALCULATION AFTER ECC ERROR
1608  001742  043040                            DH000A
1609  001744  041470                            DT0021
1610  001746  042234                            DF0042
1611
1612                                    ;       ERROR 43
1613  001750  047106                            EM5043                  ;ERROR IN ECC POSITION COUNTING AFTER ECC ERROR
1614  001752  043040                            DH000A
1615  001754  041554                            DT0043
1616  001756  042254                            DF0043
1617
1618                                    ;       ERROR 44
1619  001760  047165                            EM5044                  ;ERROR AFTER PROCESSING DATA CHECK ERR
```

```
1620  001762  044623                    E5004A
1621  001764  041322                    DT0004
1622  001766  041640                    DF0004
1623
1624                            ;       ERROR 45
1625  001770  047165                    EM5044          ;ERROR AFTER PROCESSING DATA CHECK ERR
1626  001772  044717                    E5004E
1627  001774  041566                    DT0045
1628  001776  042274                    DF0045
1629
1630                            ;       ERROR 46
1631  002000  044557                    EM5046          ;TRANSFER LENGTH ERROR PARTIAL SEC READ
1632  002002  043040                    DH000A
1633  002004  041576                    DT0046
1634  002006  042320                    DF0046
1635
1636                            ;       ERROR 47
1637  002010  047327                    EM5047          ;BSE DID NOT PREVENT DA OR DC INCREMENT
1638  002012  045124                    E5011A
1639  002014  041432                    DT0014
1640  002016  042040                    DF0014
1641
1642                            ;       ERROR 50
1643  002020  047422                    EM5050          ;BSE DID NOT CAUSE CERR
1644  002022  044623                    E5004A
1645  002024  041322                    DT0004
1646  002026  042340                    DF0050
1647
1648                            ;       ERROR 51
1649  002030  047475                    EM5051          ;BSE FAILED TO SET WHEN EXPECTED
1650  002032  044717                    E5004E
1651  002034  041566                    DT0045
1652  002036  042274                    DF0045
1653
1654                            ;       ERROR 52
1655  002040  047552                    EM5052          ;DATA XFER ABORT TO SOON
1656  002042  043040                    DH000A
1657  002044  041576                    DT0046
1658  002046  042320                    DF0046
1659
1660                            ;       ERROR 53
1661  002050  047672                    EM5053          ;MR1 ERROR IN WRT CHK OR ECC READ AFTER WRT CHK
1662  002052  044642                    E5004B
1663  002054  041370                    DT0010
1664  002056  041764                    DF0010
1665
1666                            ;       ERROR 54
1667  002060  047761                    EM5054          ;ECC ERROR IN WRITE CHECK OP
1668  002062  045336                    E5021A
1669  002064  041470                    DT0021
1670  002066  042110                    DF0021
1671
1672                            ;       ERROR 55
1673  002070  047672                    EM5053          ;CS1 ERROR AFTER WRT CHK OR ECC READ IN WRT CHK
1674  002072  044623                    E5004A
1675  002074  041406                    DT0011
```

```
1676  002076  042010                        DF0011
1677
1678                              ;         ERROR 56
1679  002100  046556                        EM5034          ;CS1 ERROR AFTER READ DATA OR ECC
1680  002102  044623                        E5004A
1681  002104  041322                        DT0004
1682  002106  041640                        DF0004
1683
1684                              ;         ERROR 57
1685  002110  047672                        EM5053          ;CS1 ERROR AFTER WRITE CHECK OR ECC READ
1686  002112  044623                        E5004A
1687  002114  041322                        DT0004
1688  002116  041640                        DF0004
1689
1690                              ;         ERROR 60
1691  002120  050024                        EM5060          ;WCE FAILED TO SET IN 16 BIT MODE
1692  002122  043040                        DH000A
1693  002124  041524                        DT0037
1694  002126  042364                        DF0060
1695
1696                              ;         ERROR 61
1697  002130  050103                        EM5061          ;TRANSFER LENGTH PROBLEM - RKBA
1698  002132  044735                        E5004F
1699  002134  041576                        DT0046
1700  002136  042404                        DF0061
1701
1702                              ;         ERROR 62
1703  002140  050103                        EM5061          ;TRANSFER LENGTH PROBLEM - RKWC
1704  002142  044753                        E5004G
1705  002144  041606                        DT0062
1706  002146  042430                        DF0062
1707
1708                              ;         ERROR 63
1709  002150  050160                        EM5063          ;WCE FAILED TO SET IN 18 BIT MODE
1710  002152  044753                        E5004G
1711  002154  041616                        DT0063
1712  002156  042454                        DF0063
1713
1714                              .SBTTL    TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
1715  002160  000000                T.CS1:  .WORD   0        ;CONTROL AND STATUS REGISTER 1
1716  002162  000000                T.WC:   .WORD   0        ;WORD COUNT REGISTER
1717  002164  000000                T.BA:   .WORD   0        ;BUS ADDRESS REGISTER
1718  002166  000000                T.DA:   .WORD   0        ;DESIRED TRACK SECTOR REGISTER
1719  002170  000000                T.CS2:  .WORD   0        ;CONTROL AND STATUS REGISTER 2
1720  002172  000000                T.DS:   .WORD   0        ;DRIVE STATUS REGISTER
1721  002174  000000                T.ER:   .WORD   0        ;ERROR REGISTER
1722  002176  000000                T.ASOF: .WORD   0        ;ATTENTION SUMMARY AND OFFSET REGISTER
1723  002200  000000                T.DCYL: .WORD   0        ;DESIRED CYLINDER REGISTER
1724  002202  000000                T.DB:   .WORD   0        ;DATA BUFFER
1725  002204  000000                T.MR1:  .WORD   0        ;MAINTENANCE REGISTER 1
1726  002206  000000                T.MR2:  .WORD   0        ;MAINTENANCE REGISTER 2
1727  002210  000000                T.MR3:  .WORD   0        ;MAINTENANCE REGISTER 3
1728  002212  000000                T.ECPS: .WORD   0        ;ECC POSITION INFORMATION
1729  002214  000000                T.ECPT: .WORD   0        ;ECC PATTERN INFORMATION
1730  002216  000000                T.SPAR: .WORD   0        ;SPARE REGISTER
1731
```

```
1732                                    .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS
1733
1734  002220  000000                    E.CS1:  .WORD   0               ;CONTROL AND STATUS REGISTER 1
1735  002222  000000                    E.WC:   .WORD   0               ;WORD COUNT REGISTER
1736  002224  000000                    E.BA:   .WORD   0               ;BUS ADDRESS REGISTER
1737  002226  000000                    E.DA:   .WORD   0               ;DESIRED TRACK SECTOR REGISTER
1738  002230  000000                    E.CS2:  .WORD   0               ;CONTROL AND STATUS REGISTER 2
1739  002232  000000                    E.DS:   .WORD   0               ;DRIVE STATUS REGISTER
1740  002234  000000                    E.ER:   .WORD   0               ;ERROR REGISTER
1741  002236  000000                    E.ASOF: .WORD   0               ;ATTENTION SUMMARY AND OFFSET REGISTER
1742  002240  000000                    E.DCYL: .WORD   0               ;DESIRED CYLINDER REGISTER
1743  002242  000000                    E.DB:   .WORD   0               ;DATA BUFFER
1744  002244  000000                    E.MR1:  .WORD   0               ;MAINTENANCE REGISTER 1
1745  002246  000000                    E.MR2:  .WORD   0               ;MAINTENANCE REGISTER 2
1746  002250  000000                    E.MR3:  .WORD   0               ;MAINTENANCE REGISTER 3
1747  002252  000000                    E.ECPS: .WORD   0               ;ECC POSITION INFORMATION
1748  002254  000000                    E.ECPT: .WORD   0               ;ECC PATTERN INFORMATION
1749  002256  000000                    E.SPAR: .WORD   0               ;SPARE REGISTER
1750
1751                                    .SBTTL  "L" REGISTERS FOR COMMAND START
1752  002260  000000                    L.CS1:  .WORD   0               ;CONTROL AND STATUS 1
1753  002262  000000                    L.WC:   .WORD   0               ;WORD COUNT
1754  002264  000000                    L.BA:   .WORD   0               ;BUFER ADDRESS
1755  002266                            L.DA:                           ;DESIRED
1756  002266      000                   L.DS:   .BYTE   0               ;SECTOR
1757  002267      000                   L.DT:   .BYTE   0               ;TRACK
1758  002270  000000                    L.CS2:  .WORD   0               ;CONTROL AND STATUS
1759  002272  000000                    L.ASOF: .WORD   0               ;ATTENTION AND OFFSET
1760  002274  000000                    L.DCYL: .WORD   0               ;DESIRED CYLINDER
1761  002276  000000                    L.MR1:  .WORD   0               ;MAINT. REG 1
1762  002300  000000                    L.MR2:  .WORD   0               ;MAINT. REG 2
1763  002302  000000                    L.MR3:  .WORD   0               ;MAINT. REG 3
1764
1765
1766
1767                                    .SBTTL  PROGRAM DEFINED VARIABLES
1768
1769  002304  000210                    RKVEC:  .WORD   210             ;RK611 VECTOR
1770  002306  000240                    RKPRI:  .WORD   PR5             ;RK611 PRIORITY
1771  002310  000000                    TRAPPC: .WORD   0               ;PC FOR MEMORY CHECK ENABLE TRAP
1772  002312  000000                    SRTFLG: .WORD   0               ;START FLAG
1773                                                                    ;   0 = 200
1774                                                                    ;   1 = 214
1775                                                                    ;  -1 = 204
1776  002314  000000                    ERRCNT: .WORD   0               ;ERROR COUNT FOR SWITCH 12 ABORT
1777  002316  000000                    P1.BIT: .WORD   0               ;NEXT BIT IN DATA SIMULATION
1778  002320  000000                    PR.BIT: .WORD   0               ;PRESENT BIT IN DATA SIMULATION
1779  002322  000000                    M1.BIT: .WORD   0               ;PREVIOUS BIT IN DATA SIMULATION
1780  002324  000000                    M2.BIT: .WORD   0               ;BIT BEFORE PREVIOUS BIT
1781  002326  000000                    BITCNT: .WORD   0               ;BIT POSITION
1782  002330  000000                    WRDCNT: .WORD   0               ;WORD COUNT FOR NPR TRANSFER
1783  002332  000000                    MEMPAR: .WORD   0               ;MEMORY EMABLE ON FIRST 24K
1784  002334  000000                    BADPAR: .WORD   0               ;BAD PARITY FLAG
1785  002336  000000                    ECCXOR: .WORD   0               ;ECC XOR SWITCH
1786  002340  000000                    ECCHI:  .WORD   0               ;ECC HIGH STORAGE
1787  002342  000000                    ECCLO:  .WORD   0               ;ECC LOW STORAGE
```

```
1788  002344  000000                ECCSRC: .WORD   0               ;ECC SOURCE FLAG
1789  002346  000000                ECPATX: .WORD   0               ;EXPECTED PATTERN ON ERROR
1790  002350  000000                ECPOSX: .WORD   0               ;EXPECTED POSITION ON ERROR
1791  002352  000000                BSEDES: .WORD   0               ;BAD SECTOR DESIRED
1792  002354  000000                HIBITS: .WORD   0               ;HI ORDER BITS FOR 18 BIT WORD XFER
1793  002356  000000                SAVSWR: .WORD   0               ;STORAGE FOR SWITCH REGISTER
```

```
1794                                        .SBTTL  PROGRAM SETUP
1795
1796  002360  012737  000001  002312  PARM:   MOV     #1,SRTFLG       ;LOAD START FLAG FOR PARMETER START
1797  002366  000406                          BR      START1
1798
1799  002370  012737  177777  002312  RESTRT: MOV     #-1,SRTFLG      ;LOAD START FLAG FOR RESTART
1800  002376  000402                          BR      START1
1801
1802  002400  005037  002312          START:  CLR     SRTFLG          ;CLEAR START FLAG
1803  002404  000005                  START1: RESET                   ;RESET THE WHOLE SYSTEM
1804  002406  105037  001103                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
1805  002412  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK POINTER
1806  002416  012746  000340                  MOV     #PR7,-(SP)      ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
1807  002422  012746  002430                  MOV     #1$,-(SP)       ;LOAD START OF PROGRAM
1808  002426  000002                          RTI                     ;LOAD PSW
1809
1810  002430  004737  037410          1$:     JSR     PC,@#$TKINT     ;INIT KEYBOARD
1811                                  .SBTTL  INITIALIZE THE COMMON TAGS
1812                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1813  002434  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1814  002440  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1815  002442  022706  001140                  CMP     #SWR,R6 ;;DONE?
1816  002446  001374                          BNE     .-6             ;;LOOP BACK IF NO
1817  002450  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1818                                  ;;INITIALIZE A FEW VECTORS
1819  002454  012737  034756  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1820  002462  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1821  002470  012737  035666  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1822  002476  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1823  002504  012737  041226  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1824  002512  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
1825  002520  012737  041072  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1826  002526  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1827  002534  013737  024706  024700          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
1828  002542  005037  001234                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1829  002546  005037  001236                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1830  002552  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1831  002560  012737  002560  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1832  002566  012737  002566  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1833                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1834                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1835  002574  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1836  002600  012737  002634  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1837  002606  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1838  002614  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1839  002622  022777  177777  176310          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1840  002630  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1841                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1842  002632  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1843  002634  012716  002642          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1844  002640  000002                          RTI
1845  002642  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1846  002650  012737  000174  001142          MOV     #DISPREG,DISPLAY
1847  002656  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1848
1849  002662  005037  001256                  CLR     $PASS           ;;CLEAR PASS COUNT
```

```
1850  002666  132737  000200  001271          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1851  002674  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
1852  002676  012737  001272  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1853  002704                          67$:
1854  002704  005037  002314                  CLR     ERRCNT          ;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
1855                                  .SBTTL  TYPE PROGRAM NAME
1856                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1857  002710  005227  177777                  INC     #-1             ;;FIRST TIME?
1858  002714  001063                          BNE     68$             ;;BRANCH IF NO
1859  002716  022737  025042  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
1860  002724  001457                          BEQ     68$             ;;BRANCH IF YES
1861  002726  104401  002774                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1862                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1863  002732  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1864  002736  001012                          BNE     70$             ;;BRANCH IF YES
1865  002740  123727  001270  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1866  002746  001406                          BEQ     70$             ;;BRANCH IF YES
1867  002750  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1868  002756  001005                          BNE     71$             ;;BRANCH IF NO
1869  002760  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1870  002762  000403                          BR      71$
1871  002764  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1872  002772                          71$:
1873  002772  000434                          BR      68$             ;;GET OVER THE ASCIZ
1874                                  ;;69$:  .ASCIZ  <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 5  MAINDEC-11-DZR6EA/<CRLF>
1875  003064                          68$:
1876  003064  005227  177777                  INC     #-1             ;TEST IF PASS 0
1877  003070  001002                          BNE     4$              ;NO - SKIP
1878  003072  104401  042612                  TYPE    ,OPR006         ;TYPE PASS TIME MESSAGE
1879  003076  022737  000001  002312  4$:     CMP     #1,SRTFLG       ;CHECK IF PARAMETER START
1880  003104  001121                          BNE     15$             ;NO, CONTINUE SETUP
1881  003106  104401  042500          5$:     TYPE    ,OPR001         ;TYPE "RK611 BUS ADDRESS (  ) ="
1882  003112  013746  001324                  MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
1883  003116  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1884  003120  104401  042532                  TYPE    ,OPR002
1885  003124  104412                          RDOCT                   ;GET VALUE
1886  003126  012637  001202                  MOV     (SP)+,$TMP0
1887  003132  001407                          BEQ     7$              ;CHECK IF <CR>
1888  003134  022737  160000  001202          CMP     #160000,$TMP0   ;CHECK IF IN I/O PAGE
1889  003142  101361                          BHI     5$
1890  003144  013737  001202  001324          MOV     $TMP0,$BASE     ;LOAD NEW BUS ADDRESS
1891  003152  104401  042540          7$:     TYPE    ,OPR003         ;TYPE "RK611 VECTOR ADDRESS (  ) ="
1892  003156  013746  001320                  MOV     $VECT1,-(SP)    ;TYPE OUT VECTOR ADDRESS
1893  003162  042716  160000                  BIC     #160000,(SP)
1894  003166  104401  042532                  TYPE    -,OPR002
1895  003172  104412                          RDOCT                   ;GET VALUE
1896  003174  012637  001202                  MOV     (SP)+,$TMP0
1897  003200  001412                          BEQ     10$             ;CHECK IF <CR>
1898  003202  022737  001000  001202          CMP     #1000,$TMP0     ;CHECK IF LEGAL
1899  003210  101760                          BLOS    7$
1900  003212  042737  017777  001320          BIC     #17777,$VECT1   ;LOAD NEW VECTOR ADDRESS
1901  003220  053737  001202  001320          BIS     $TMP0,$VECT1
1902  003226  104401  042570          10$:    TYPE    ,OPR004         ;TYPE "RK611 PRIORITY (  ) ="
1903  003232  005046                          CLR     -(SP)
1904  003234  113716  001321                  MOVB    $VECT1+1,(SP)
1905  003240  006216                          ASR     (SP)            ;SHIFT 5 BITS RIGHT
```

```
1906  003242  006216                          ASR     (SP)
1907  003244  006216                          ASR     (SP)
1908  003246  006216                          ASR     (SP)
1909  003250  006216                          ASR     (SP)
1910  003252  104402                          TYPOC
1911  003254  104401  042532                 TYPE    ,OPR002
1912  003260  104412                          RDOCT                   ;GET VALUE
1913  003262  012637  001202                 MOV     (SP)+,$TMP0
1914  003266  001430                          BEQ     15$             ;CHECK FOR DEFAULT
1915  003270  022737  000007  001202         CMP     #7,$TMP0        ;CHECK IF LEGAL
1916  003276  103753                          BLO     10$
1917  003300  022737  000004  001202         CMP     #4,$TMP0
1918  003306  101347                          BHI     10$
1919  003310  006337  001202                 ASL     $TMP0           ;SHIFT 5 BITS LEFT
1920  003314  006337  001202                 ASL     $TMP0
1921  003320  006337  001202                 ASL     $TMP0
1922  003324  006337  001202                 ASL     $TMP0
1923  003330  006337  001202                 ASL     $TMP0
1924  003334  042737  160000  001320         BIC     #160000,$VECT1  ;STORE NEW PRIORITY
1925  003342  153737  001202  001321         BISB    $TMP0,$VECT1+1
1926  003350  013737  001320  002304  15$:    MOV     $VECT1,RKVEC    ;STORE RK611 VECTOR
1927  003356  042737  160000  002304         BIC     #160000,RKVEC
1928  003364  113737  001321  002306         MOVB    $VECT1+1,RKPRI  ;STORE PRIORITY
1929  003372  004737  034450                 JSR     PC,$SIZE        ;SIZE MEMORY
1930
1931  003376  013702  001324         NEWPAS: MOV     $BASE,R2        ;SET RK POINTER
1932  003402  005037  001236                 CLR     $ESCAPE         ;CLEAR ESCAPE
1933  003406  004737  034314                 JSR     PC,PARCHK       ;CHECK OF MEMORY CHECK ENABLE
1934  003412  012746  000000                 MOV     #PR0,-(SP)      ;LOCK OUT INTERRUPTS
1935  003416  012746  003424                 MOV     #TST1,-(SP)
1936  003422  000002                          RTI
```

```
                                .SBTTL  **MULTI-SECTOR WRITE OPERATIONS

                                ;;*****************************************************************
                                ;*TEST 1         MULTI-SECTOR WRITE (PART 1)
                                ;*
                                ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                ;*      DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                                ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
                                ;*      TWO ECC WORDS.  MAKE SURE THE NEXT HEADER RECOGNITION
                                ;*      CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR.
                                ;*      PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION
                                ;*      OF HEADER.
                                ;*
                                ;;*****************************************************************
003424  000004                  TST1:   SCOPE
003426  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS

003434  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLR CONTROLLER

003442  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED

003446  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
003452  000000                          0                       ;CYLINDER 0
003454     000                          .BYTE   0               ;SECTOR 0
003455     000                          .BYTE   0               ;TRACK 0
003456  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
003460  177377                          -401                    ;WORD COUNT -401
003462  000023                          WRDATA                  ;COMMAND WRDATA

003464  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
003470  000001                          1                       ;PATTERN 1

003472  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION

003476  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
003502  104004                          ERROR   4               ;CS1 MISCOMPARE
003504  104005                          ERROR   5               ;MR1     "
003506  104006                          ERROR   6               ;MR2     "
003510  104007                          ERROR   7               ;MR3     "

003512  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

003516  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
003522  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
003524  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR

003526  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
003532  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
003534  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
003536  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

003540  012737  045211  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
003546  004437  027002                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
```

```
1993  003552  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
1994  003554  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
1995  003556  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
1996  003560  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
1997
1998  003562  012737  045270  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
1999  003570  004437  027402                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2000  003574  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2001  003576  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2002  003600  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2003
2004  003602  012700  000034                  MOV     #7*4,R0         ;ISSUE CLOCKS TO NEXT HDR COMPARE
2005  003606  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2006  003614  012762  000040  000026          MOV     #DMD,RKMR1(R2)
2007  003622  005300                          DEC     R0
2008  003624  001370                          BNE     1$
2009
2010  003626  013703  002274                  MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2011  003632  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2012  003636  010337  002246                  MOV     R3,E.MR2        ;STORE EXPECTED MR2
2013  003642  113703  002267                  MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2014  003646  042703  177400                  BIC     #177400,R3      ;CLEAR UNUSED BITS
2015  003652  012700  000005                  MOV     #5,R0           ;SET COUNT FOR SHIFTING
2016  003656  006303                  5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2017  003660  005300                          DEC     R0              ;DEC COUNT
2018  003662  001375                          BNE     5$              ;LOOP UNTIL ZERO
2019  003664  153703  002266                  BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2020
2021  003670  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2022  003676  001402                          BEQ     6$              ;NO - SKIP
2023  003700  052703  001000                  BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2024  003704                          6$:
2025  003704  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2026  003710  010337  002250                  MOV     R3,E.MR3        ;STORE EXPECTED MR3
2027  003714  016237  000034  002206          MOV     RKMR2(R2),T.MR2 ;GET MR2
2028  003722  016237  000036  002210          MOV     RKMR3(R2),T.MR3 ;GET MR3
2029  003730  016237  000000  002160          MOV     RKCS1(R2),T.CS1 ;GET CS1
2030  003736  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2031  003744  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF OK
2032  003752  001402                          BEQ     3$              ;YES - SKIP
2033  003754  104025                          ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2034  003756  000427                          BR      TST2            ;;GO TO NEXT TEST
2035
2036  003760  023737  002206  002246  3$:     CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2037  003766  001402                          BEQ     2$              ;YES - SKIP
2038  003770  104023                          ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2039  003772  000421                          BR      TST2            ;;GO TO NEXT TEST
2040
2041  003774  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2042  004002  001402                          BEQ     4$              ;YES - SKIP
2043  004004  104024                          ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2044  004006  000413                          BR      TST2            ;;GO TO NEXT TEST
2045
2046  004010  004437  025062          4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2047
2048  004014  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
```

```
2049  004020  104010                        ERROR   10              ;MR1 ERROR
2050  004022  104011                        ERROR   11              ;CS1 ERROR
2051
2052  004024  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2053  004030  104012                        ERROR   12              ;MR1 ERROR
2054  004032  104013                        ERROR   13              ;CS1 ERROR
2055  004034  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR
2056
2057                                ;;*****************************************************************
2058                                ;*TEST 2        MULTI-SECTOR WRITE (PART 2)
2059                                ;*
2060                                ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2061                                ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2062                                ;*      DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2063                                ;*      CYLINDER 0, HEAD 0, SECTOR 23.  CLOCK THROUGH SEEK
2064                                ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2065                                ;*      AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
2066                                ;*      TWO ECC WORDS.  MAKE SURE THE NEXT HEADER RECOGNITION
2067                                ;*      CYCLE STARTS AND THE PROPER HEADER IS SEARCHED FOR.
2068                                ;*      PROVIDE A SECOND GOOD HEADER AND CHECK RECOGNITION
2069                                ;*      OF HEADER.
2070                                ;*
2071                                ;;*****************************************************************
2072  004036  000004                TST2:   SCOPE
2073  004040  012737  000012  001234        MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2074
2075  004046  012762  100000  000000        MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2076
2077  004054  005037  002352                CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2078
2079  004060  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2080  004064  000000                        0                       ;CYLINDER 0
2081  004066      023                       .BYTE   23              ;SECTOR 23
2082  004067      000                       .BYTE   0               ;TRACK 0
2083  004070  051444                        OBUFF                   ;BUFFER ADDRESS OBUFF
2084  004072  177377                        -401                    ;WORD COUNT -401
2085  004074  000023                        WRDATA                  ;COMMAND WRDATA
2086
2087  004076  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
2088  004102  000002                        2                       ;PATTERN 2
2089
2090  004104  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
2091
2092  004110  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2093  004114  104004                        ERROR   4               ;CS1 MISCOMPARE
2094  004116  104005                        ERROR   5               ;MR1     "
2095  004120  104006                        ERROR   6               ;MR2     "
2096  004122  104007                        ERROR   7               ;MR3     "
2097
2098  004124  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2099
2100  004130  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2101  004134  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
2102  004136  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
2103
2104  004140  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
```

```
2105  004144  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
2106  004146  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2107  004150  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2108
2109  004152  012737  045211  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2110  004160  004437  027002                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2111  004164  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
2112  004166  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2113  004170  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2114  004172  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2115
2116  004174  012737  045270  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2117  004202  004437  027402                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2118  004206  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2119  004210  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2120  004212  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2121
2122  004214  012700  000034                  MOV     #7*4,R0         ;ISSUE CLOCKS TO NEXT HDR COMPARE
2123  004220  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2124  004226  012762  000040  000026          MOV     #DMD,RKMR1(R2)
2125  004234  005300                          DEC     R0
2126  004236  001370                          BNE     1$
2127
2128  004240  013703  002274                  MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2129  004244  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2130  004250  010337  002246                  MOV     R3,E.MR2        ;STORE EXPECTED MR2
2131  004254  113703  002267                  MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2132  004260  042703  177400                  BIC     #177400,R3      ;CLEAR UNUSED BITS
2133  004264  012700  000005                  MOV     #5,R0           ;SET COUNT FOR SHIFTING
2134  004270  006303                  5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2135  004272  005300                          DEC     R0              ;DEC COUNT
2136  004274  001375                          BNE     5$              ;LOOP UNTIL ZERO
2137  004276  153703  002266                  BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2138
2139  004302  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2140  004310  001402                          BEQ     6$              ;NO - SKIP
2141  004312  052703  001000                  BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2142  004316                          6$:
2143  004316  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2144  004322  010337  002250                  MOV     R3,E.MR3        ;STORE EXPECTED MR3
2145  004326  016237  000034  002206          MOV     RKMR2(R2),T.MR2 ;GET MR2
2146  004334  016237  000036  002210          MOV     RKMR3(R2),T.MR3 ;GET MR3
2147  004342  016237  000000  002160          MOV     RKCS1(R2),T.CS1 ;GET CS1
2148  004350  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2149  004356  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF OK
2150  004364  001402                          BEQ     3$              ;YES - SKIP
2151  004366  104025                          ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2152  004370  000427                          BR      TST3            ;;GO TO NEXT TEST
2153
2154  004372  023737  002206  002246  3$:     CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2155  004400  001402                          BEQ     2$              ;YES - SKIP
2156  004402  104023                          ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2157  004404  000421                          BR      TST3            ;;GO TO NEXT TEST
2158
2159  004406  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2160  004414  001402                          BEQ     4$              ;YES - SKIP
```

```
2161  004416  104024                          ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2162  004420  000413                          BR      TST3            ;;GO TO NEXT TEST
2163
2164  004422  004437  025062          4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2165
2166  004426  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2167  004432  104010                          ERROR   10              ;MR1 ERROR
2168  004434  104011                          ERROR   11              ;CS1 ERROR
2169
2170  004436  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2171  004442  104012                          ERROR   12              ;MR1 ERROR
2172  004444  104013                          ERROR   13              ;CS1 ERROR
2173  004446  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR
2174
2175                                  ;;*****************************************************************
2176                                  ;*TEST 3        MID-TRANSFER SEEK (PART 1)
2177                                  ;*
2178                                  ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2179                                  ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2180                                  ;*      DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2181                                  ;*      CYLINDER 0, HEAD 0, SECTOR 25.  CLOCK THROUGH SEEK
2182                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2183                                  ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
2184                                  ;*      TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
2185                                  ;*      AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2186                                  ;*      MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2187                                  ;*
2188                                  ;;*****************************************************************
2189  004450  000004                  TST3:   SCOPE
2190  004452  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2191
2192  004460  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2193
2194  004466  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2195
2196  004472  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2197  004476  000000                          0                       ;CYLINDER 0
2198  004500     025                          .BYTE   25              ;SECTOR 25
2199  004501     000                          .BYTE   0               ;TRACK 0
2200  004502  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2201  004504  177377                          -401                    ;WORD COUNT -401
2202  004506  000023                          WRDATA                  ;COMMAND WRDATA
2203
2204  004510  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2205  004514  000003                          3                       ;PATTERN 3
2206
2207  004516  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2208
2209  004522  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2210  004526  104004                          ERROR   4               ;CS1 MISCOMPARE
2211  004530  104005                          ERROR   5               ;MR1     "
2212  004532  104006                          ERROR   6               ;MR2     "
2213  004534  104007                          ERROR   7               ;MR3     "
2214
2215  004536  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2216
```

```
2217  004542  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2218  004546  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
2219  004550  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
2220
2221  004552  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2222  004556  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
2223  004560  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2224  004562  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2225
2226  004564  012737  045211  001510        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2227  004572  004437  027002                JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2228  004576  104015                        ERROR   15              ;MR1 IN ERROR IN READ GAP
2229  004600  104016                        ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2230  004602  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2231  004604  104020                        ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2232
2233  004606  012737  045270  001510        MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2234  004614  004437  027402                JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2235  004620  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2236  004622  104021                        ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2237  004624  104022                        ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2238
2239  004626  012700  000324                MOV     #53.*4,R0               ;ISSUE CLOCKS TO NEXT HDR COMPARE
2240  004632  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2241  004640  012762  000040  000026        MOV     #DMD,RKMR1(R2)
2242  004646  005300                        DEC     R0
2243  004650  001370                        BNE     1$
2244
2245  004652  013703  002274                MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2246  004656  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2247  004662  010337  002246                MOV     R3,E.MR2        ;STORE EXPECTED MR2
2248  004666  113703  002267                MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2249  004672  042703  177400                BIC     #177400,R3      ;CLEAR UNUSED BITS
2250  004676  012700  000005                MOV     #5,R0           ;SET COUNT FOR SHIFTING
2251  004702  006303                5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2252  004704  005300                        DEC     R0              ;DEC COUNT
2253  004706  001375                        BNE     5$              ;LOOP UNTIL ZERO
2254  004710  153703  002266                BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2255
2256  004714  032737  010000  002260        BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2257  004722  001402                        BEQ     6$              ;NO - SKIP
2258  004724  052703  001000                BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2259  004730                        6$:
2260  004730  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2261  004734  010337  002250                MOV     R3,E.MR3        ;STORE EXPECTED MR3
2262  004740  016237  000034  002206        MOV     RKMR2(R2),T.MR2 ;GET MR2
2263  004746  016237  000036  002210        MOV     RKMR3(R2),T.MR3 ;GET MR3
2264  004754  016237  000000  002160        MOV     RKCS1(R2),T.CS1 ;GET CS1
2265  004762  013737  002260  002220        MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2266  004770  023737  002160  002220        CMP     T.CS1,E.CS1     ;CHECK IF OK
2267  004776  001402                        BEQ     3$              ;YES - SKIP
2268  005000  104025                        ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2269  005002  000427                        BR      TST4            ;;GO TO NEXT TEST
2270
2271  005004  023737  002206  002246  3$:   CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2272  005012  001402                        BEQ     2$              ;YES - SKIP
```

```
2273  005014  104023                          ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2274  005016  000421                          BR      TST4            ;;GO TO NEXT TEST
2275
2276  005020  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2277  005026  001402                          BEQ     4$              ;YES - SKIP
2278  005030  104024                          ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2279  005032  000413                          BR      TST4            ;;GO TO NEXT TEST
2280
2281  005034  004437  025062          4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2282
2283  005040  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2284  005044  104010                          ERROR   10              ;MR1 ERROR
2285  005046  104011                          ERROR   11              ;CS1 ERROR
2286
2287  005050  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2288  005054  104012                          ERROR   12              ;MR1 ERROR
2289  005056  104013                          ERROR   13              ;CS1 ERROR
2290  005060  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR
2291
2292                                  ;;*****************************************************************
2293                                  ;*TEST 4         MID-TRANSFER SEEK (PART 2)
2294                                  ;*
2295                                  ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2296                                  ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2297                                  ;*      DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2298                                  ;*      CYLINDER 0, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
2299                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2300                                  ;*      AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
2301                                  ;*      TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
2302                                  ;*      AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2303                                  ;*      MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2304                                  ;*
2305                                  ;;*****************************************************************
2306  005062  000004          TST4:   SCOPE
2307  005064  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2308
2309  005072  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2310
2311  005100  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2312
2313  005104  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2314  005110  000000                          0                       ;CYLINDER 0
2315  005112     025                          .BYTE   25              ;SECTOR 25
2316  005113     002                          .BYTE   2               ;TRACK 2
2317  005114  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2318  005116  177377                          -401                    ;WORD COUNT -401
2319  005120  000023                          WRDATA                  ;COMMAND WRDATA
2320
2321  005122  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2322  005126  000004                          4                       ;PATTERN 4
2323
2324  005130  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2325
2326  005134  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2327  005140  104004                          ERROR   4               ;CS1 MISCOMPARE
2328  005142  104005                          ERROR   5               ;MR1     "
```

```
2329  005144  104006                        ERROR   6               ;MR2     "
2330  005146  104007                        ERROR   7               ;MR3     "
2331
2332  005150  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2333
2334  005154  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2335  005160  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
2336  005162  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
2337
2338  005164  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2339  005170  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
2340  005172  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2341  005174  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2342
2343  005176  012737  045211  001510        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2344  005204  004437  027002                JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2345  005210  104015                        ERROR   15              ;MR1 IN ERROR IN READ GAP
2346  005212  104016                        ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2347  005214  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2348  005216  104020                        ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2349
2350  005220  012737  045270  001510        MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2351  005226  004437  027402.               JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2352  005232  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2353  005234  104021                        ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2354  005236  104022                        ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2355
2356  005240  012700  000324                MOV     #53.*4,R0               ;ISSUE CLOCKS TO NEXT HDR COMPARE
2357  005244  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2358  005252  012762  000040  000026        MOV     #DMD,RKMR1(R2)
2359  005260  005300                        DEC     R0
2360  005262  001370                        BNE     1$
2361
2362  005264  013703  002274                MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2363  005270  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2364  005274  010337  002246                MOV     R3,E.MR2        ;STORE EXPECTED MR2
2365  005300  113703  002267                MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2366  005304  042703  177400                BIC     #177400,R3      ;CLEAR UNUSED BITS
2367  005310  012700  000005                MOV     #5,R0           ;SET COUNT FOR SHIFTING
2368  005314  006303                5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2369  005316  005300                        DEC     R0              ;DEC COUNT
2370  005320  001375                        BNE     5$              ;LOOP UNTIL ZERO
2371  005322  153703  002266                BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2372
2373  005326  032737  010000  002260        BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2374  005334  001402                        BEQ     6$              ;NO - SKIP
2375  005336  052703  001000                BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2376  005342                        6$:
2377  005342  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2378  005346  010337  002250                MOV     R3,E.MR3        ;STORE EXPECTED MR3
2379  005352  016237  000034  002206        MOV     RKMR2(R2),T.MR2 ;GET MR2
2380  005360  016237  000036  002210        MOV     RKMR3(R2),T.MR3 ;GET MR3
2381  005366  016237  000000  002160        MOV     RKCS1(R2),T.CS1 ;GET CS1
2382  005374  013737  002260  002220        MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2383  005402  023737  002160  002220        CMP     T.CS1,E.CS1     ;CHECK IF OK
2384  005410  001402                        BEQ     3$              ;YES - SKIP
```

```
2385  005412  104025                          ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2386  005414  000427                          BR      TST5            ;;GO TO NEXT TEST
2387
2388  005416  023737  002206  002246  3$:     CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2389  005424  001402                          BEQ     2$              ;YES - SKIP
2390  005426  104023                          ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2391  005430  000421                          BR      TST5            ;;GO TO NEXT TEST
2392
2393  005432  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2394  005440  001402                          BEQ     4$              ;YES - SKIP
2395  005442  104024                          ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2396  005444  000413                          BR      TST5            ;;GO TO NEXT TEST
2397
2398  005446  004437  025062          4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2399
2400  005452  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2401  005456  104010                          ERROR   10              ;MR1 ERROR
2402  005460  104011                          ERROR   11              ;CS1 ERROR
2403
2404  005462  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2405  005466  104012                          ERROR   12              ;MR1 ERROR
2406  005470  104013                          ERROR   13              ;CS1 ERROR
2407  005472  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR
2408
2409                          ;;*****************************************************************
2410                          ;*TEST 5        MID-TRANSFER SEEK (PART 3)
2411                          ;*
2412                          ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2413                          ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2414                          ;*      DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
2415                          ;*      CYLINDER 0, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
2416                          ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2417                          ;*      AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
2418                          ;*      TWO WORD ECC. MAKE SURE THE CONTROLLER ISSUES
2419                          ;*      AN IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2420                          ;*      MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2421                          ;*
2422                          ;*****************************************************************
2423  005474  000004          TST5:   SCOPE
2424  005476  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2425
2426  005504  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2427
2428  005512  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2429
2430  005516  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2431  005522  000000                          0                       ;CYLINDER 0
2432  005524      025                         .BYTE   25              ;SECTOR 25
2433  005525      002                         .BYTE   2               ;TRACK 2
2434  005526  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2435  005530  177377                          -401                    ;WORD COUNT -401
2436  005532  002023                          WRDATA!CDT                      ;COMMAND WRDATA!CDT
2437
2438  005534  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2439  005540  000005                          5                       ;PATTERN 5
2440
```

```
2441  005542  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2442
2443  005546  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2444  005552  104004                          ERROR   4               ;CS1 MISCOMPARE
2445  005554  104005                          ERROR   5               ;MR1     "
2446  005556  104006                          ERROR   6               ;MR2     "
2447  005560  104007                          ERROR   7               ;MR3     "
2448
2449  005562  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2450
2451  005566  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2452  005572  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
2453  005574  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
2454
2455  005576  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2456  005602  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
2457  005604  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2458  005606  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2459
2460  005610  012737  045211  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2461  005616  004437  027002                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2462  005622  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
2463  005624  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2464  005626  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2465  005630  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2466
2467  005632  012737  045270  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2468  005640  004437  027402                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2469  005644  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2470  005646  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2471  005650  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2472
2473  005652  012700  000324                  MOV     #53.*4,R0               ;ISSUE CLOCKS TO NEXT HDR COMPARE
2474  005656  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2475  005664  012762  000040  000026          MOV     #DMD,RKMR1(R2)
2476  005672  005300                          DEC     R0
2477  005674  001370                          BNE     1$
2478
2479  005676  013703  002274                  MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2480  005702  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2481  005706  010337  002246                  MOV     R3,E.MR2        ;STORE EXPECTED MR2
2482  005712  113703  002267                  MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2483  005716  042703  177400                  BIC     #177400,R3      ;CLEAR UNUSED BITS
2484  005722  012700  000005                  MOV     #5,R0           ;SET COUNT FOR SHIFTING
2485  005726  006303                  5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2486  005730  005300                          DEC     R0              ;DEC COUNT
2487  005732  001375                          BNE     5$              ;LOOP UNTIL ZERO
2488  005734  153703  002266                  BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2489
2490  005740  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2491  005746  001402                          BEQ     6$              ;NO - SKIP
2492  005750  052703  001000                  BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2493  005754                           6$:
2494  005754  004437  034242                  JSR     R4,FSBLVV       ;INVERT WORD
2495  005760  010337  002250                  MOV     R3,E.MR3        ;STORE EXPECTED MR3
2496  005764  016237  000034  002206          MOV     RKMR2(R2),T.MR2 ;GET MR2
```

```
2497  005772  016237  000036  002210          MOV     RKMR3(R2),T.MR3 ;GET MR3
2498  006000  016237  000000  002160          MOV     RKCS1(R2),T.CS1 ;GET CS1
2499  006006  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2500  006014  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF OK
2501  006022  001402                          BEQ     3$              ;YES - SKIP
2502  006024  104025                          ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2503  006026  000427                          BR      TST6            ;;GO TO NEXT TEST
2504
2505  006030  023737  002206  002246  3$:     CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2506  006036  001402                          BEQ     2$              ;YES - SKIP
2507  006040  104023                          ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2508  006042  000421                          BR      TST6            ;;GO TO NEXT TEST
2509
2510  006044  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2511  006052  001402                          BEQ     4$              ;YES - SKIP
2512  006054  104024                          ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2513  006056  000413                          BR      TST6            ;;GO TO NEXT TEST
2514
2515  006060  004437  025062          4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2516
2517  006064  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2518  006070  104010                          ERROR   10              ;MR1 ERROR
2519  006072  104011                          ERROR   11              ;CS1 ERROR
2520
2521  006074  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2522  006100  104012                          ERROR   12              ;MR1 ERROR
2523  006102  104013                          ERROR   13              ;CS1 ERROR
2524  006104  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR
2525
2526                                  ;;*****************************************************************
2527                                  ;*TEST 6        MID-TRANSFER SEEK (PART 4)
2528                                  ;*
2529                                  ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2530                                  ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2531                                  ;*      DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
2532                                  ;*      CYLINDER 0, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK AND
2533                                  ;*      DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2534                                  ;*      AND A GOOD HEADER, CLOCK THROUGH 400 WORDS AND THE
2535                                  ;*      TWO WORD ECC.  MAKE SURE THE CONTROLLER ISSUES
2536                                  ;*      IMPLIED SEEK TO THE RIGHT HEAD AND CYLINDER.
2537                                  ;*      MAKE SURE THE NEXT HEADER RECOGNITION PROCEEDS PROPERLY.
2538                                  ;*
2539                                  ;;*****************************************************************
2540  006106  000004                  TST6:   SCOPE
2541  006110  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2542
2543  006116  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2544
2545  006124  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2546
2547  006130  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2548  006134  000000                          0                       ;CYLINDER 0
2549  006136     025                          .BYTE   25              ;SECTOR 25
2550  006137     004                          .BYTE   4               ;TRACK 4
2551  006140  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2552  006142  177377                          -401                    ;WORD COUNT -401
```

```
2553  006144  002023                        WRDATA!CDT                      ;COMMAND WRDATA!CDT
2554
2555  006146  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
2556  006152  000006                        6                       ;PATTERN 6
2557
2558  006154  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
2559
2560  006160  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2561  006164  104004                        ERROR   4               ;CS1 MISCOMPARE
2562  006166  104005                        ERROR   5               ;MR1     "
2563  006170  104006                        ERROR   6               ;MR2     "
2564  006172  104007                        ERROR   7               ;MR3     "
2565
2566  006174  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2567
2568  006200  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2569  006204  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
2570  006206  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
2571
2572  006210  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2573  006214  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
2574  006216  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2575  006220  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2576
2577  006222  012737  045211  001510        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2578  006230  004437  027002                JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2579  006234  104015                        ERROR   15              ;MR1 IN ERROR IN READ GAP
2580  006236  104016                        ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2581  006240  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2582  006242  104020                        ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2583
2584  006244  012737  045270  001510        MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2585  006252  004437  027402                JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2586  006256  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2587  006260  104021                        ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2588  006262  104022                        ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2589
2590  006264  012700  000324                MOV     #53.*4,R0               ;ISSUE CLOCKS TO NEXT HDR COMPARE
2591  006270  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2592  006276  012762  000040  000026        MOV     #DMD,RKMR1(R2)
2593  006304  005300                        DEC     R0
2594  006306  001370                        BNE     1$
2595
2596  006310  013703  002274                MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2597  006314  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2598  006320  010337  002246                MOV     R3,E.MR2        ;STORE EXPECTED MR2
2599  006324  113703  002267                MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2600  006330  042703  177400                BIC     #177400,R3      ;CLEAR UNUSED BITS
2601  006334  012700  000005                MOV     #5,R0           ;SET COUNT FOR SHIFTING
2602  006340  006303                5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2603  006342  005300                        DEC     R0              ;DEC COUNT
2604  006344  001375                        BNE     5$              ;LOOP UNTIL ZERO
2605  006346  153703  002266                BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2606
2607  006352  032737  010000  002260        BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2608  006360  001402                        BEQ     6$              ;NO - SKIP
```

```
2609  006362  052703  001000              BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2610  006366                      6$:
2611  006366  004437  034242              JSR     R4,FSBLVV       ;INVERT WORD
2612  006372  010337  002250              MOV     R3,E.MR3        ;STORE EXPECTED MR3
2613  006376  016237  000034  002206      MOV     RKMR2(R2),T.MR2 ;GET MR2
2614  006404  016237  000036  002210      MOV     RKMR3(R2),T.MR3 ;GET MR3
2615  006412  016237  000000  002160      MOV     RKCS1(R2),T.CS1 ;GET CS1
2616  006420  013737  002260  002220      MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2617  006426  023737  002160  002220      CMP     T.CS1,E.CS1     ;CHECK IF OK
2618  006434  001402                      BEQ     3$              ;YES - SKIP
2619  006436  104025                      ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2620  006440  000427                      BR      TST7            ;;GO TO NEXT TEST
2621
2622  006442  023737  002206  002246  3$: CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2623  006450  001402                      BEQ     2$              ;YES - SKIP
2624  006452  104023                      ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2625  006454  000421                      BR      TST7            ;;GO TO NEXT TEST
2626
2627  006456  023737  002210  002250  2$: CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2628  006464  001402                      BEQ     4$              ;YES - SKIP
2629  006466  104024                      ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2630  006470  000413                      BR      TST7            ;;GO TO NEXT TEST
2631
2632  006472  004437  025062          4$: JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2633
2634  006476  004437  025504              JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2635  006502  104010                      ERROR   10              ;MR1 ERROR
2636  006504  104011                      ERROR   11              ;CS1 ERROR
2637
2638  006506  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2639  006512  104012                      ERROR   12              ;MR1 ERROR
2640  006514  104013                      ERROR   13              ;CS1 ERROR
2641  006516  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR
2642
2643                          ;;*****************************************************************
2644                          ;*TEST 7        MID-TRANSFER SEEK AND 24 SECTOR FORMAT
2645                          ;*
2646                          ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2647                          ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE DATA
2648                          ;*      DATA OF 401 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
2649                          ;*      CYLINDER 0, HEAD 0, SECTOR 23.  CLOCK THROUGH SEEK
2650                          ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2651                          ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 18 BIT WORDS
2652                          ;*      AND THE 32 BIT ECC.  CHECK THAT IMPLIED SEEK IS
2653                          ;*      ISSUE TO NEXT TRACK.  CHECK FOR PROPER HEADER RECOGNITION
2654                          ;*      OCCURS.
2655                          ;*
2656                          ;;*****************************************************************
2657  006520  000004          TST7:   SCOPE
2658  006522  012737  000012  001234      MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2659
2660  006530  012762  100000  000000      MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2661
2662  006536  005037  002352              CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2663  006542  012700  002000              MOV     #2000,R0        ;SET A COUNT TO DELAY FOR CLEAR
2664  006546  005300                  20$: DEC     R0
```

```
2665  006550  001376                        BNE     20$             ;LOOP UNTIL ZERO
2666
2667  006552  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2668  006556  000000                        0                       ;CYLINDER 0
2669  006560     023                        .BYTE   23              ;SECTOR 23
2670  006561     000                        .BYTE   0               ;TRACK 0
2671  006562  051444                        OBUFF                   ;BUFFER ADDRESS OBUFF
2672  006564  177377                        -401                    ;WORD COUNT -401
2673  006566  010023                        WRDATA!CFMT                     ;COMMAND WRDATA!CFMT
2674
2675  006570  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
2676  006574  000007                        7                       ;PATTERN 7
2677
2678  006576  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
2679
2680  006602  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2681  006606  104004                        ERROR   4               ;CS1 MISCOMPARE
2682  006610  104005                        ERROR   5               ;MR1     "
2683  006612  104006                        ERROR   6               ;MR2     "
2684  006614  104007                        ERROR   7               ;MR3     "
2685
2686  006616  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2687
2688  006622  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2689  006626  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
2690  006630  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
2691
2692  006632  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2693  006636  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
2694  006640  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2695  006642  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2696
2697  006644  012737  045211  001510        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2698  006652  004437  027002                JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2699  006656  104015                        ERROR   15              ;MR1 IN ERROR IN READ GAP
2700  006660  104016                        ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2701  006662  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2702  006664  104020                        ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2703
2704  006666  012737  045270  001510        MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2705  006674  004437  027402                JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2706  006700  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2707  006702  104021                        ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2708  006704  104022                        ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2709
2710  006706  012700  000324                MOV     #53.*4,R0               ;ISSUE CLOCKS TO NEXT HDR COMPARE
2711  006712  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK LOOP
2712  006720  012762  000040  000026        MOV     #DMD,RKMR1(R2)
2713  006726  005300                        DEC     R0
2714  006730  001370                        BNE     1$
2715
2716  006732  013703  002274                MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
2717  006736  004437  034242                JSR     R4,FSBLVV       ;INVERT WORD
2718  006742  010337  002246                MOV     R3,E.MR2        ;STORE EXPECTED MR2
2719  006746  113703  002267                MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
2720  006752  042703  177400                BIC     #177400,R3      ;CLEAR UNUSED BITS
```

```
2721  006756  012700  000005            MOV     #5,R0           ;SET COUNT FOR SHIFTING
2722  006762  006303            5$:     ASL     R3              ;SHIFT TRACK FOR HEADER ALIGNMENT
2723  006764  005300                    DEC     R0              ;DEC COUNT
2724  006766  001375                    BNE     5$              ;LOOP UNTIL ZERO
2725  006770  153703  002266            BISB    L.DS,R3         ;INSERT SECTOR  NUMBER
2726
2727  006774  032737  010000  002260    BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
2728  007002  001402                    BEQ     6$              ;NO - SKIP
2729  007004  052703  001000            BIS     #BIT9,R3        ;SET FORMAT BIT FOR WORD TWO
2730  007010                    6$:
2731  007010  004437  034242            JSR     R4,FSBLVV       ;INVERT WORD
2732  007014  010337  002250            MOV     R3,E.MR3        ;STORE EXPECTED MR3
2733  007020  016237  000034  002206    MOV     RKMR2(R2),T.MR2 ;GET MR2
2734  007026  016237  000036  002210    MOV     RKMR3(R2),T.MR3 ;GET MR3
2735  007034  016237  000000  002160    MOV     RKCS1(R2),T.CS1 ;GET CS1
2736  007042  013737  002260  002220    MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
2737  007050  023737  002160  002220    CMP     T.CS1,E.CS1     ;CHECK IF OK
2738  007056  001402                    BEQ     3$              ;YES - SKIP
2739  007060  104025                    ERROR   25              ;CS1 IN ERROR AT MULTI-SECTOR TIME
2740  007062  000427                    BR      TST10           ;;GO TO NEXT TEST
2741
2742  007064  023737  002206  002246  3$:     CMP     T.MR2,E.MR2     ;CHECK IF MR2 OK
2743  007072  001402                    BEQ     2$              ;YES - SKIP
2744  007074  104023                    ERROR   23              ;MR2 IN ERROR AT MULTI-SECTOR TIME
2745  007076  000421                    BR      TST10           ;;GO TO NEXT TEST
2746
2747  007100  023737  002210  002250  2$:     CMP     T.MR3,E.MR3     ;CHECK IF MR3 OK
2748  007106  001402                    BEQ     4$              ;YES - SKIP
2749  007110  104024                    ERROR   24              ;MR3 IN ERROR AT MULTI-SECTOR TIME
2750  007112  000413                    BR      TST10           ;;GO TO NEXT TEST
2751
2752  007114  004437  025062    4$:     JSR     R4,DSECTR       ;SIMULATE 2ND SECTOR PULSE
2753
2754  007120  004437  025504            JSR     R4,DRSYNC       ;SIMULATE PREAMBLE
2755  007124  104010                    ERROR   10              ;MR1 ERROR
2756  007126  104011                    ERROR   11              ;CS1 ERROR
2757
2758  007130  004437  026042            JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2759  007134  104012                    ERROR   12              ;MR1 ERROR
2760  007136  104013                    ERROR   13              ;CS1 ERROR
2761  007140  104014                    ERROR   14              ;RKDCYL OR RKDA IN ERROR
2762
2763                    ;;*****************************************************************
2764                    ;*TEST 10       CYLINDER OVERFLOW (PART 1)
2765                    ;*
2766                    ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2767                    ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2768                    ;*      DATA OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2769                    ;*      CYLINDER 632, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
2770                    ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2771                    ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND THE
2772                    ;*      TWO WORD ECC.  MAKE SURE CYLINDER OVERFLOW ERROR DOES
2773                    ;*      NOT SET.
2774                    ;*
2775                    ;;*****************************************************************
2776  007142  000004    TST10:  SCOPE
```

```
2777  007144  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2778
2779  007152  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2780  007156  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2781
2782  007164  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2783  007170  000632                          632                     ;CYLINDER 632
2784  007172     025                          .BYTE   25              ;SECTOR 25
2785  007173     002                          .BYTE   2               ;TRACK 2
2786  007174  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2787  007176  177400                          -400                    ;WORD COUNT -400
2788  007200  000023                          WRDATA                  ;COMMAND WRDATA
2789
2790  007202  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2791  007206  000001                          1                       ;PATTERN 1
2792
2793  007210  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2794
2795  007214  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2796  007220  104004                          ERROR   4               ;CS1 MISCOMPARE
2797  007222  104005                          ERROR   5               ;MR1     "
2798  007224  104006                          ERROR   6               ;MR2     "
2799  007226  104007                          ERROR   7               ;MR3     "
2800
2801  007230  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2802
2803  007234  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2804  007240  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
2805  007242  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
2806
2807  007244  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2808  007250  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
2809  007252  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2810  007254  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2811
2812  007256  012737  045211  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2813  007264  004437  027002                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2814  007270  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
2815  007272  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2816  007274  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2817  007276  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2818
2819  007300  012737  045270  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2820  007306  004437  027402                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2821  007312  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2822  007314  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2823  007316  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2824
2825
2826  007320  012700  000016                  MOV     #3*4+2,R0       ;SET COUNT TO END OPERATION
2827  007324  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;SET CLOCK
2828  007332  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;CLEAR CLOCK
2829  007340  005300                          DEC     R0              ;DEC COUNT
2830  007342  001370                          BNE     1$              ;LOOP UNTIL 0
2831
2832  007344  004437  034154                  JSR     R4,GETREG       ;GET RK611 REGISTERS
```

```
2833  007350  032737  100000  002160          BIT     #CERR,T.CS1     ;TEST IF ANY ERROR SET
2834  007356  001415                          BEQ     3$              ;NO - SKIP TO EXIT
2835
2836  007360  032737  001000  002174          BIT     #COE,T.ER       ;TEST IF IT IS COE
2837  007366  001405                          BEQ     2$              ;NO - SKIP
2838  007370  012737  046061  001622          MOV     #E5026A,EH030   ;SET MESSAGE HEADER
2839  007376  104030                          ERROR   30              ;"UNEXPECTED COE ERROR"
2840  007400  000404                          BR      3$              ;EXIT
2841
2842  007402  012737  046061  001632  2$:     MOV     #E5026A,EH031   ;SET HEADER
2843  007410  104031                          ERROR   31              ;"UNEXPECTED ERROR TESTING COE"
2844
2845  007412                          3$:
2846
2847                                  ;;*****************************************************************
2848                                  ;*TEST 11       CYLINDER OVERFLOW (PART 2)
2849                                  ;*
2850                                  ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2851                                  ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2852                                  ;*      DATA OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
2853                                  ;*      CYLINDER 632, HEAD 2, SECTOR 25.  CLOCK THROUGH SEEK
2854                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2855                                  ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
2856                                  ;*      TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.
2857                                  ;*
2858                                  ;;*****************************************************************
2859  007412  000004                  TST11:  SCOPE
2860  007414  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2861
2862  007422  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2863  007426  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2864
2865  007434  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2866  007440  000632                          632                     ;CYLINDER 632
2867  007442     025                          .BYTE   25              ;SECTOR 25
2868  007443     002                          .BYTE   2               ;TRACK 2
2869  007444  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2870  007446  177377                          -401                    ;WORD COUNT -401
2871  007450  000023                          WRDATA                  ;COMMAND WRDATA
2872
2873  007452  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2874  007456  000002                          2                       ;PATTERN 2
2875
2876  007460  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2877
2878  007464  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2879  007470  104004                          ERROR   4               ;CS1 MISCOMPARE
2880  007472  104005                          ERROR   5               ;MR1     "
2881  007474  104006                          ERROR   6               ;MR2     "
2882  007476  104007                          ERROR   7               ;MR3     "
2883
2884  007500  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2885
2886  007504  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2887  007510  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
2888  007512  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
```

```
2889
2890  007514  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2891  007520  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
2892  007522  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2893  007524  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2894
2895  007526  012737  045211  001510        MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2896  007534  004437  027002                JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2897  007540  104015                        ERROR   15              ;MR1 IN ERROR IN READ GAP
2898  007542  104016                        ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2899  007544  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2900  007546  104020                        ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2901
2902  007550  012737  045270  001510        MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2903  007556  004437  027402                JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2904  007562  104017                        ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2905  007564  104021                        ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2906  007566  104022                        ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2907
2908
2909  007570  012700  000016                MOV     #3*4+2,R0       ;SET COUNT TO COMPLETE OPERATION
2910  007574  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2)  ;RUN THE CLOCK
2911  007602  012762  000040  000026        MOV     #DMD,RKMR1(R2)
2912  007610  005300                        DEC     R0              ;DEC COUNT
2913  007612  001370                        BNE     1$              ;LOOP UNTIL 0
2914
2915  007614  004437  034154                JSR     R4,GETREG       ;GET RK611 REGS
2916  007620  032737  001000  002174        BIT     #COE,T.ER       ;TEST IF COE SET
2917  007626  001012                        BNE     3$              ;YES - EXIT TEST
2918
2919  007630  032737  100000  002160        BIT     #CERR,T.CS1     ;TEST IF ANY OTHER ERROR SET
2920  007636  001402                        BEQ     2$              ;NO - SKIP
2921  007640  104027                        ERROR   27              ;"NO COE WHEN EXPECTED"
2922  007642  000404                        BR      3$              ;GO TO EXIT
2923
2924  007644  012737  046061  001602  2$:   MOV     #E5026A,EH026   ;SET HEADER
2925  007652  104026                        ERROR   26              ;"UNEXPECTED ERROR WHEN FORCING COE"
2926
2927  007654                          3$:
2928
2929                                        ;;*****************************************************************
2930                                        ;*TEST 12       CYLINDER OVERFLOW (PART 3)
2931                                        ;*
2932                                        ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
2933                                        ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
2934                                        ;*      DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
2935                                        ;*      CYLINDER 632, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK
2936                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
2937                                        ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
2938                                        ;*      TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW.
2939                                        ;*      ERROR DOES NOT SET.
2940                                        ;*
2941                                        ;;*****************************************************************
2942  007654  000004                  TST12: SCOPE
2943  007656  012737  000012  001234        MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
2944
```

```
2945  007664  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
2946
2947  007672  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
2948
2949  007676  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
2950  007702  000632                          632                     ;CYLINDER 632
2951  007704      025                         .BYTE   25              ;SECTOR 25
2952  007705      004                         .BYTE   4               ;TRACK 4
2953  007706  051444                          OBUFF                   ;BUFFER ADDRESS OBUFF
2954  007710  177377                          -401                    ;WORD COUNT -401
2955  007712  002023                          WRDATA!CDT                      ;COMMAND WRDATA!CDT
2956
2957  007714  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
2958  007720  000003                          3                       ;PATTERN 3
2959
2960  007722  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
2961
2962  007726  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
2963  007732  104004                          ERROR   4               ;CS1 MISCOMPARE
2964  007734  104005                          ERROR   5               ;MR1     "
2965  007736  104006                          ERROR   6               ;MR2     "
2966  007740  104007                          ERROR   7               ;MR3     "
2967
2968  007742  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
2969
2970  007746  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
2971  007752  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
2972  007754  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
2973
2974  007756  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
2975  007762  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
2976  007764  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
2977  007766  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
2978
2979  007770  012737  045211  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
2980  007776  004437  027002                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
2981  010002  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
2982  010004  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
2983  010006  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
2984  010010  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
2985
2986  010012  012737  045270  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
2987  010020  004437  027402                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
2988  010024  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
2989  010026  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
2990  010030  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
2991
2992
2993  010032  012700  000016                  MOV     #3*4+2,R0       ;SET COUNT TO END OPERATION
2994  010036  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;SET CLOCK
2995  010044  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;CLEAR CLOCK
2996  010052  005300                          DEC     R0              ;DEC COUNT
2997  010054  001370                          BNE     1$              ;LOOP UNTIL 0
2998
2999  010056  004437  034154                  JSR     R4,GETREG       ;GET RK611 REGISTERS
3000  010062  032737  100000  002160          BIT     #CERR,T.CS1     ;TEST IF ANY ERROR SET
```

```
3001  010070  001415                          BEQ     3$                ;NO - SKIP TO EXIT
3002  010072  032737  001000  002174          BIT     #COE,T.ER         ;CHECK IF COE
3003  010100  001405                          BEQ     2$                ;NO - SKIP
3004
3005  010102  012737  046232  001622          MOV     #E50268,EH030     ;SET HEADER
3006  010110  104030                          ERROR   30                ;"UNEXPECTED COE ERROR"
3007  010112  000404                          BR      3$                ;EXIT
3008
3009  010114  012737  046232  001632  2$:     MOV     #E50268,EH031     ;SET HEADER
3010  010122  104031                          ERROR   31                ;"UNEXPECTED ERROR WHILE TESTING COE"
3011
3012  010124                          3$:
3013
3014                                  ;;*****************************************************************
3015                                  ;*TEST 13       CYLINDER OVERFLOW (PART 4)
3016                                  ;*
3017                                  ;*      CLEAR THE RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3018                                  ;*      PUT THE CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
3019                                  ;*      DATA OF 401 WORDS IN 26 SECTOR FORMAT WITH CDT SET,
3020                                  ;*      CYLINDER 1456, HEAD 4, SECTOR 25.  CLOCK THROUGH SEEK
3021                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
3022                                  ;*      AND A GOOD HEADER.  CLOCK THROUGH 400 WORDS AND
3023                                  ;*      TWO ECC WORDS. MAKE SURE CYLINDER OVERFLOW ERROR SETS.
3024                                  ;*
3025                                  ;;*****************************************************************
3026  010124  000004                  TST13:  SCOPE
3027  010126  012737  000012  001234          MOV     #10.,$TIMES       ;;DO 10. ITERATIONS
3028
3029  010134  005037  002352                  CLR     BSEDES            ;CLEAR BAD SEC DESIRED
3030  010140  012762  100000  000000          MOV     #CCLR,RKCS1(R2)   ;CLEAR CONTROLLER
3031
3032  010146  004437  034026                  JSR     R4,LOADRK         ;LOAD "L" REGISTERS
3033  010152  001456                          1456                      ;CYLINDER 1456
3034  010154      025                         .BYTE   25                ;SECTOR 25
3035  010155      004                         .BYTE   4                 ;TRACK 4
3036  010156  051444                          OBUFF                     ;BUFFER ADDRESS OBUFF
3037  010160  177377                          -401                      ;WORD COUNT -401
3038  010162  002023                          WRDATA!CDT                        ;COMMAND WRDATA!CDT
3039
3040  010164  004437  033612                  JSR     R4,BLDDAT         ;GO BUILD DATA
3041  010170  000004                          4                         ;PATTERN 4
3042
3043  010172  004437  034072                  JSR     R4,OPSTRT         ;START THE OPERATION
3044
3045  010176  004437  025100                  JSR     R4,DISEEK         ;SIMULATE IMPLIED SEEK
3046  010202  104004                          ERROR   4                 ;CS1 MISCOMPARE
3047  010204  104005                          ERROR   5                 ;MR1     "
3048  010206  104006                          ERROR   6                 ;MR2     "
3049  010210  104007                          ERROR   7                 ;MR3     "
3050
3051  010212  004437  025062                  JSR     R4,DSECTR         ;SIMULATE SECTOR PULSE
3052
3053  010216  004437  025504                  JSR     R4,DRSYNC         ;SIMULATE HEADER PREAMBLE
3054  010222  104010                          ERROR   10                ;MR1 CONTENTS IN ERROR
3055  010224  104011                          ERROR   11                ;CS1 CONTENTS IN ERROR
3056
```

```
3057  010226  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3058  010232  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
3059  010234  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3060  010236  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3061
3062  010240  012737  045211  001510      MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
3063  010246  004437  027002              JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
3064  010252  104015                      ERROR   15              ;MR1 IN ERROR IN READ GAP
3065  010254  104016                      ERROR   16              ;CS1 IN ERROR AFTER READ GAP
3066  010256  104017                      ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
3067  010260  104020                      ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC
3068
3069  010262  012737  045270  001510      MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
3070  010270  004437  027402              JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
3071  010274  104017                      ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
3072  010276  104021                      ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
3073  010300  104022                      ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC
3074
3075
3076  010302  012700  000016              MOV     #3*4+2,R0       ;SET COUNT TO END OPERATION
3077  010306  012762  000440  000026  1$: MOV     #DMD!MCLK,RKMR1(R2) ;SET CLOCK
3078  010314  012762  000040  000026      MOV     #DMD,RKMR1(R2)  ;CLEAR CLOCK
3079  010322  005300                      DEC     R0              ;DEC COUNT
3080  010324  001370                      BNE     1$              ;LOOP UNTIL 0
3081
3082  010326  004437  034154              JSR     R4,GETREG       ;GET RK611 REGISTERS
3083  010332  032737  001000  002174      BIT     #COE,T.ER       ;TEST IF COE SET
3084  010340  001012                      BNE     3$              ;YES - EXIT
3085
3086  010342  032737  100000  002160      BIT     #CERR,T.CS1     ;TEST IF ANY OTHER ERROR SET
3087  010350  001402                      BEQ     2$              ;NO SKIP
3088  010352  104027                      ERROR   27              ;"UNEXPECTED ERROR WHILE TESTING COE"
3089  010354  000404                      BR      3$              ;EXIT
3090
3091  010356  012737  046232  001602  2$: MOV     #E5026B,EH026   ;SET HEADER
3092  010364  104026                      ERROR   26              ;"COE DID NOT SET WHEN EXPECTED"
3093  010366                          3$:
3094
3095                                  .SBTTL  **NPR WRITING OF MEMORY
3096
3097                                  ;;*****************************************************************
3098                                  ;*TEST 14       NPR WRITING OF MEMORY
3099                                  ;*
3100                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3101                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3102                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3103                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3104                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3105                                  ;*      A GOOD HEADER, 40 DATA WORDS.  VERIFY THAT THE WORDS
3106                                  ;*      ARE WRITTEN PROPERLY IN MEMORY.
3107                                  ;*
3108                                  ;;*****************************************************************
3109  010366  000004                  TST14:  SCOPE
3110  010370  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
3111
3112  010376  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
```

```
3113  010402  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3114
3115  010410  005037  002344                  CLR     ECCSRC          ;CLEAR ECC SOURCE
3116  010414  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
3117  010420  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3118  010424  000000                          0                       ;CYLINDER 0
3119  010426     000                          .BYTE   0               ;SECTOR 0
3120  010427     000                          .BYTE   0               ;TRACK 0
3121  010430  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3122  010432  177400                          -400                    ;WORD COUNT -400
3123  010434  000021                          RDDATA                  ;COMMAND RDDATA
3124
3125  010436  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3126
3127  010442  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
3128  010446  000003                          3                       ;PATTERN 3
3129
3130
3131  010450  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
3132  010454  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3133  010460  104004                          ERROR   4               ;CS1 MISCOMPARE
3134  010462  104005                          ERROR   5               ;MR1     "
3135  010464  104006                          ERROR   6               ;MR2     "
3136  010466  104007                          ERROR   7               ;MR3     "
3137
3138  010470  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3139
3140  010474  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3141  010500  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
3142  010502  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
3143
3144  010504  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3145  010510  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
3146  010512  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3147  010514  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3148
3149  010516  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3150  010522  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
3151  010524  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3152  010526  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
3153  010530  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3154
3155  010532  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
3156  010540  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3157  010546  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
3158  010552  000060                          60                      ;NUMBER OF WORDS TO SIMULATE
3159  010554  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3160  010556  104035                          ERROR   35              ;ECC ERROR READING DATA
3161  010560  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3162  010562  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3163  010564  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
3164  010566  104043                          ERROR   43              ;ERR IN ECC POS COUNT
3165
3166
3167  010570  012700  051444                  MOV     #OBUFF,R0       ;SET POINTER TO GOOD DATA
3168  010574  012701  050440                  MOV     #IBUFF,R1       ;SET POINTER TO INPUT DATA
```

```
3169
3170  010600  012704  000012              MOV     #10.,R4         ;SET ERROR LIMIT COUNT
3171  010604  005037  001236              CLR     $ESCAPE         ;CLEAR ESCAPE
3172  010610  005037  001220              CLR     $TMP7           ;CLEAR ERROR REPORT SWITCH
3173  010614  012703  000040              MOV     #40,R3          ;SET COMPARE LIMIT
3174  010620  005005                      CLR     R5              ;CLEAR WORD COUNTER
3175
3176  010622  021011              1$:     CMP     (R0),(R1)       ;COMPARE DATA
3177  010624  001005                      BNE     2$              ;SKIP IF NOT EQUAL
3178  010626  005303              4$:     DEC     R3              ;ELSE DEC WORD COUNT LIMIT
3179  010630  001424                      BEQ     5$              ;IF 0 - SKIP TO EXIT
3180  010632  005205                      INC     R5              ;BUMP WORD COUNT
3181  010634  022021                      CMP     (R0)+,(R1)+     ;BUMP DATA POINTERS
3182  010636  000771                      BR      1$              ;LOOP
3183
3184  010640  005304              2$:     DEC     R4              ;DEC ERROR LIMIT
3185  010642  001417                      BEQ     5$              ;IF 0 - SKIP
3186
3187  010644  011037  001162              MOV     (R0),$REG0      ;GET GOOD WORD FOR REPORT
3188  010650  011137  001164              MOV     (R1),$REG1      ;GET BAD WORD
3189  010654  010537  001166              MOV     R5,$REG2        ;GET WORD NUMBER
3190
3191  010660  005737  001220              TST     $TMP7           ;DECIDE WHICH ERROR REPORT TO USE
3192  010664  001004                      BNE     3$              ;SKIP IF NOT 0
3193  010666  005237  001220              INC     $TMP7           ;ELSE BUMP $TMP7
3194  010672  104037                      ERROR   37              ;REPORT FIRST ERROR LINE
3195  010674  000754                      BR      4$              ;SKIP
3196
3197  010676  104040              3$:     ERROR 40                ;REPORT ALL OTHER LINES
3198  010700  000752                      BR      4$              ;SKIP
3199
3200  010702                      5$:
3201
3202                                      .SBTTL  **ECC ERROR DETECTION/CORRECTION TESTS
3203
3204                                      ;;*****************************************************************
3205                                      ;*TEST 15       READ DATA WITH NO ERROR
3206                                      ;*
3207                                      ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3208                                      ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3209                                      ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3210                                      ;*      CLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3211                                      ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3212                                      ;*      A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
3213                                      ;*      VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.
3214                                      ;*
3215                                      ;;*****************************************************************
3216  010702  000004              TST15:  SCOPE
3217  010704  012737  000001  001234      MOV     #1,$TIMES       ;;DO 1 ITERATION
3218
3219  010712  005037  002352              CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3220  010716  012762  100000  000000      MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3221  010724  005037  002344              CLR     ECCSRC          ;CLEAR ECC SOURCE FLAG
3222  010730  005037  002344              CLR     ECCSRC          ;CLEAR ECC SOURCE
3223  010734  005037  002354              CLR     HIBITS          ;CLEAR HI ORDER BITS
3224  010740  004437  034026              JSR     R4,LOADRK       ;LOAD "L" REGISTERS
```

```
3225  010744  000000                          0                       ;CYLINDER 0
3226  010746     000                          .BYTE   0               ;SECTOR 0
3227  010747     000                          .BYTE   0               ;TRACK 0
3228  010750  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3229  010752  177400                          -400                    ;WORD COUNT -400
3230  010754  000021                          RDDATA                  ;COMMAND RDDATA
3231
3232  010756  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3233
3234  010762  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
3235  010766  000004                          4                       ;PATTERN 4
3236
3237
3238  010770  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
3239  010774  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3240  011000  104004                          ERROR   4               ;CS1 MISCOMPARE
3241  011002  104005                          ERROR   5               ;MR1     "
3242  011004  104006                          ERROR   6               ;MR2     "
3243  011006  104007                          ERROR   7               ;MR3     "
3244
3245  011010  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3246
3247  011014  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3248  011020  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
3249  011022  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
3250
3251  011024  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3252  011030  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
3253  011032  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3254  011034  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3255
3256  011036  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3257  011042  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
3258  011044  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3259  011046  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
3260  011050  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3261
3262  011052  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
3263  011060  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3264  011066  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
3265  011072  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
3266  011074  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3267  011076  104035                          ERROR   35              ;ECC ERROR READING DATA
3268  011100  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3269  011102  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3270  011104  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
3271  011106  104043                          ERROR   43              ;ERR IN ECC POS COUNT
3272
3273
3274  011110  012700  002710                  MOV     #40.*37.,R0     ;SET COUNT TO EMPTY SILO
3275
3276  011114  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
3277  011122  012762  000040  000026          MOV     #DMD,RKMR1(R2)
3278  011130  005300                          DEC     R0              ;DEC COUNT
3279  011132  001370                          BNE     1$              ;LOOP UNTIL 0
3280
```

```
3281  011134  016237  000000  002160          MOV     RKCS1(R2),T.CS1 ;GET CS1
3282  011142  004437  034154                  JSR     R4,GETREG       ;GET 611 REGS
3283  011146  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO BIT
3284  011154  052737  000200  002220          BIS     #RDY,E.CS1      ;SET READY BIT
3285  011162  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3286  011170  001401                          BEQ     2$              ;YES - SKIP
3287  011172  104056                          ERROR   56              ;"CS1 IN ERROR AFTER READ DATA"
3288  011174                          2$:
3289                                  ;;*****************************************************************
3290                                  ;*TEST 16       READ DATA WITH ONE BIT BURST ERROR (PART 1)
3291                                  ;*
3292                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3293                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3294                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3295                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3296                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3297                                  ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
3298                                  ;*      A ONE BIT BURST ERROR OF A 0 THAT SHOULD BE A 1.
3299                                  ;*      VERIFY THE COUNTING OF THE POSITION
3300                                  ;*      REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
3301                                  ;*      MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
3302                                  ;*
3303                                  ;;*****************************************************************
3304  011174  000004                  TST16:  SCOPE
3305  011176  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
3306
3307  011204  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3308  011210  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3309
3310  011216  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
3311  011222  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3312  011226  000000                          0                       ;CYLINDER 0
3313  011230     000                          .BYTE   0               ;SECTOR 0
3314  011231     000                          .BYTE   0               ;TRACK 0
3315  011232  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3316  011234  177400                          -400                    ;WORD COUNT -400
3317  011236  000021                          RDDATA                  ;COMMAND RDDATA
3318
3319  011240  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3320
3321  011244  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
3322  011250  000007                          7                       ;PATTERN 7
3323
3324  011252  012700  000530                  MOV     #530,R0         ;SET INDEX INTO BUFFER
3325
3326  011256  012737  177446  052444          MOV     #177446,OBUFF+1000      ;STORE ECC WORDS
3327  011264  012737  015457  052446          MOV     #15457,OBUFF+1002
3328  011272  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
3329  011300  032760  000100  051444          BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
3330  011306  001004                          BNE     100$            ;YES - SKIP
3331  011310  052760  000100  051444          BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
3332  011316  000403                          BR      101$
3333  011320  042760  000100  051444  100$:   BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
3334  011326                          101$:
3335
3336  011326  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
```

```
3337  011332  004437  025100                    JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3338  011336  104004                            ERROR   4               ;CS1 MISCOMPARE
3339  011340  104005                            ERROR   5               ;MR1     "
3340  011342  104006                            ERROR   6               ;MR2     "
3341  011344  104007                            ERROR   7               ;MR3     "
3342
3343  011346  004437  025062                    JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3344
3345  011352  004437  025504                    JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3346  011356  104010                            ERROR   10              ;MR1 CONTENTS IN ERROR
3347  011360  104011                            ERROR   11              ;CS1 CONTENTS IN ERROR
3348
3349  011362  004437  026042                    JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3350  011366  104012                            ERROR   12              ;MR1 CONTENTS IN ERROR
3351  011370  104013                            ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3352  011372  104014                            ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3353
3354  011374  004437  030276                    JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3355  011400  104015                            ERROR   15              ;MR1 IN ERROR READING GAP
3356  011402  104016                            ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3357  011404  104032                            ERROR   32              ;MR1 IN ERROR READING SYNC
3358  011406  104033                            ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3359
3360  011410  012737  000001  002346            MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
3361  011416  012737  004066  002350            MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3362  011424  004437  030774                    JSR     R4,DREAD        ;GO SIMULATE DATA READ
3363  011430  000400                            400                     ;NUMBER OF WORDS TO SIMULATE
3364  011432  104034                            ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3365  011434  104035                            ERROR   35              ;ECC ERROR READING DATA
3366  011436  104036                            ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3367  011440  104041                            ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3368  011442  104042                            ERROR   42              ;ERR IN ECC PAT CALC.
3369  011444  104043                            ERROR   43              ;ERR IN ECC POS COUNT
3370
3371
3372  011446  012737  000020  002326            MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3373  011454  012700  000005                    MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
3374  011460  012701  013672                    MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3375  011464  005037  001214                    CLR     $TMP5           ;CLEAR SWITCH
3376
3377  011470  004737  033414          3$:       JSR     PC,RDBIT        ;GO READ BIT
3378  011474  004737  032260                    JSR     PC,ECCGEN       ;GENERATE ECC
3379  011500  016237  000032  002214            MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
3380  011506  016237  000030  002212            MOV     RKECPS(R2),T.ECPS       ;GET POSITION
3381  011514  005237  002350                    INC     ECPOSX          ;BUMP POSITION EXPECTED
3382  011520  013737  002350  002252            MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
3383  011526  023737  002254  002214            CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
3384  011534  001401                            BEQ     1$              ;YES - SKIP
3385  011536  104042                            ERROR   42
3386  011540  023737  002252  002212    1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
3387  011546  001401                            BEQ     2$              ;YES - SKIP
3388  011550  104043                            ERROR   43
3389  011552  005237  002326            2$:     INC     BITCNT          ;BUMP BIT COUNT
3390  011556  005301                            DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
3391  011560  001343                            BNE     3$              ;NOT YET ZERO - LOOP
3392  011562  005300                            DEC     R0              ;DEC PASS THRU 0 COUNT
```

```
3393  011564  001406                          BEQ     4$              ;IF 0 - SKIP
3394  011566  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
3395  011574  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
3396  011600  000733                          BR      3$              ;GO DO LOOP AGAIN
3397
3398  011602  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
3399  011606  001012                          BNE     5$              ;ALREADY SET - SKIP
3400  011610  012701  005276                  MOV     #5276,R1        ;SET COUNT TO FINAL POSITION
3401  011614  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
3402  011620  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
3403  011626  005237  001214                  INC     $TMP5           ;BUMP SWITCH
3404  011632  000716                          BR      3$              ;GO DO LOOP AGAIN
3405
3406  011634  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
3407  011640  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
3408  011644  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
3409  011652  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
3410  011660  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
3411  011666  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3412  011674  001402                          BEQ     6$              ;YES - SKIP
3413  011676  104044                          ERROR   44
3414  011700  000410                          BR      7$
3415  011702  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
3416  011710  001404                          BEQ     7$              ;NO - SKIP
3417  011712  012737  100000  002234          MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
3418  011720  104045                          ERROR   45
3419  011722                          7$:
3420
3421                                  ;;**********************************************************************
3422                                  ;*TEST 17       READ DATA WITH 11 BIT BURST ERROR
3423                                  ;*
3424                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3425                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE. ISSUE A READ DATA
3426                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3427                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0. CLOCK THROUGH SEEK
3428                                  ;*      AND DRIVE CLEAR MESSAGES. SIMULATE A SECTOR PULSE,
3429                                  ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
3430                                  ;*      AN 11 BIT BURST ERROR. VERIFY THE COUNTING OF THE POSITION
3431                                  ;*      REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
3432                                  ;*      MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
3433                                  ;*
3434                                  ;;**********************************************************************
3435  011722  000004          TST17:  SCOPE
3436  011724  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
3437  011732  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3438  011736  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3439
3440  011744  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
3441  011750  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3442  011754  000000                          0                       ;CYLINDER 0
3443  011756     000                          .BYTE   0               ;SECTOR 0
3444  011757     000                          .BYTE   0               ;TRACK 0
3445  011760  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3446  011762  177400                          -400                    ;WORD COUNT -400
3447  011764  000021                          RDDATA                  ;COMMAND RDDATA
3448
```

```
3449  011766  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3450
3451  011772  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
3452  011776  000007                        7                       ;PATTERN 7
3453
3454  012000  012700  000530                MOV     #530,R0         ;SET INDEX INTO BUFFER
3455
3456  012004  012737  177446  052444        MOV     #177446,OBUFF+1000      ;STORE ECC WORDS
3457  012012  012737  015457  052446        MOV     #15457,OBUFF+1002
3458  012020  012737  000001  002344        MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
3459  012026  032760  000100  051444        BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
3460  012034  001004                        BNE     100$            ;YES - SKIP
3461  012036  052760  000100  051444        BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
3462  012044  000403                        BR      101$
3463  012046  042760  000100  051444  100$: BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
3464  012054                          101$:
3465
3466  012054  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
3467  012060  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3468  012064  104004                        ERROR   4               ;CS1 MISCOMPARE
3469  012066  104005                        ERROR   5               ;MR1     "
3470  012070  104006                        ERROR   6               ;MR2     "
3471  012072  104007                        ERROR   7               ;MR3     "
3472
3473  012074  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3474
3475  012100  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3476  012104  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
3477  012106  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
3478
3479  012110  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3480  012114  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
3481  012116  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3482  012120  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3483
3484  012122  004437  030276                JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3485  012126  104015                        ERROR   15              ;MR1 IN ERROR READING GAP
3486  012130  104016                        ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3487  012132  104032                        ERROR   32              ;MR1 IN ERROR READING SYNC
3488  012134  104033                        ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3489
3490  012136  012737  000001  002346        MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
3491  012144  012737  004066  002350        MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3492  012152  004437  030774                JSR     R4,DREAD        ;GO SIMULATE DATA READ
3493  012156  000400                        400                     ;NUMBER OF WORDS TO SIMULATE
3494  012160  104034                        ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3495  012162  104035                        ERROR   35              ;ECC ERROR READING DATA
3496  012164  104036                        ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3497  012166  104041                        ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3498  012170  104042                        ERROR   42              ;ERR IN ECC PAT CALC.
3499  012172  104043                        ERROR   43              ;ERR IN ECC POS COUNT
3500
3501
3502  012174  012737  000020  002326        MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3503  012202  012700  000005                MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
3504  012206  012701  013672                MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
```

```
3505  012212  005037  001214                CLR     $TMP5           ;CLEAR SWITCH
3506
3507  012216  004737  033414        3$:     JSR     PC,RDBIT        ;GO READ BIT
3508  012222  004737  032260                JSR     PC,ECCGEN       ;GENERATE ECC
3509  012226  016237  000032  002214        MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
3510  012234  016237  000030  002212        MOV     RKECPS(R2),T.ECPS       ;GET POSITION
3511  012242  005237  002350                INC     ECPOSX          ;BUMP POSITION EXPECTED
3512  012246  013737  002350  002252        MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
3513  012254  023737  002254  002214        CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
3514  012262  001401                        BEQ     1$              ;YES - SKIP
3515  012264  104042                        ERROR   42
3516  012266  023737  002252  002212  1$:   CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
3517  012274  001401                        BEQ     2$              ;YES - SKIP
3518  012276  104043                        ERROR   43
3519  012300  005237  002326        2$:     INC     BITCNT          ;BUMP BIT COUNT
3520  012304  005301                        DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
3521  012306  001343                        BNE     3$              ;NOT YET ZERO - LOOP
3522  012310  005300                        DEC     R0              ;DEC PASS THRU 0 COUNT
3523  012312  001406                        BEQ     4$              ;IF 0 - SKIP
3524  012314  012737  177777  002350        MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
3525  012322  012701  020000                MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
3526  012326  000733                        BR      3$              ;GO DO LOOP AGAIN
3527
3528  012330  005737  001214        4$:     TST     $TMP5           ;TEST SWITCH
3529  012334  001012                        BNE     5$              ;ALREADY SET - SKIP
3530  012336  012701  005276                MOV     #5276,R1        ;SET COUNT TO FINAL POSITION
3531  012342  012700  000001                MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
3532  012346  012737  177777  002350        MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
3533  012354  005237  001214                INC     $TMP5           ;BUMP SWITCH
3534  012360  000716                        BR      3$              ;GO DO LOOP AGAIN
3535
3536  012362  004737  033414        5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
3537  012366  004437  034154                JSR     R4,GETREG       ;GET '611 REGISTERS
3538  012372  013737  002260  002220        MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
3539  012400  052737  100200  002220        BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
3540  012406  042737  000001  002220        BIC     #GO,E.CS1       ;CLEAR GO
3541  012414  023737  002160  002220        CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3542  012422  001402                        BEQ     6$              ;YES - SKIP
3543  012424  104044                        ERROR   44
3544  012426  000410                        BR      7$
3545  012430  032737  000100  002174  6$:   BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
3546  012436  001404                        BEQ     7$              ;NO SKIP
3547  012440  012737  100000  002234        MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
3548  012446  104045                        ERROR   45
3549
3550  012450                        7$:
3551
3552                                ;;*****************************************************************
3553                                ;*TEST 20       READ DATA WITH ONE BIT BURST ERROR (PART 2)
3554                                ;*
3555                                ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3556                                ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3557                                ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3558                                ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3559                                ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3560                                ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
```

```
3561                                    ;*      A ONE BIT BURST ERROR OF A 1 THAT SHOULD BE A 0.
3562                                    ;*      VERIFY THE COUNTING OF THE POSITION
3563                                    ;*      REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
3564                                    ;*      MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
3565                                    ;*
3566                                    ;;*****************************************************************
3567  012450  000004                    TST20:  SCOPE
3568  012452  012737  000001  001234            MOV     #1,$TIMES       ;;DO 1 ITERATION
3569
3570  012460  005037  002352            CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3571  012464  012762  100000  000000    MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3572
3573  012472  005037  002354            CLR     HIBITS          ;CLEAR HI ORDER BITS
3574  012476  004437  034026            JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3575  012502  000000                    0                       ;CYLINDER 0
3576  012504     000                    .BYTE   0               ;SECTOR 0
3577  012505     000                    .BYTE   0               ;TRACK 0
3578  012506  050440                    IBUFF                   ;BUFFER ADDRESS IBUFF
3579  012510  177400                    -400                    ;WORD COUNT -400
3580  012512  000021                    RDDATA                  ;COMMAND RDDATA
3581
3582  012514  004437  033566            JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3583
3584  012520  004437  033612            JSR     R4,BLDDAT       ;GO BUILD DATA
3585  012524  000005                    5                       ;PATTERN 5
3586
3587  012526  012700  000530            MOV     #530,R0         ;SET INDEX INTO BUFFER
3588
3589  012532  012737  126073  052444    MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
3590  012540  012737  151052  052446    MOV     #151052,OBUFF+1002
3591  012546  012737  000001  002344    MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
3592  012554  032760  000100  051444    BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
3593  012562  001004                    BNE     100$            ;YES - SKIP
3594  012564  052760  000100  051444    BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
3595  012572  000403                    BR      101$
3596  012574  042760  000100  051444  100$:  BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
3597  012602                          101$:
3598
3599  012602  004437  034072            JSR     R4,OPSTRT       ;START THE OPERATION
3600  012606  004437  025100            JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3601  012612  104004                    ERROR   4               ;CS1 MISCOMPARE
3602  012614  104005                    ERROR   5               ;MR1     "
3603  012616  104006                    ERROR   6               ;MR2     "
3604  012620  104007                    ERROR   7               ;MR3     "
3605
3606  012622  004437  025062            JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3607
3608  012626  004437  025504            JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3609  012632  104010                    ERROR   10              ;MR1 CONTENTS IN ERROR
3610  012634  104011                    ERROR   11              ;CS1 CONTENTS IN ERROR
3611
3612  012636  004437  026042            JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3613  012642  104012                    ERROR   12              ;MR1 CONTENTS IN ERROR
3614  012644  104013                    ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3615  012646  104014                    ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3616
```

```
3617  012650  004437  030276              JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3618  012654  104015                      ERROR   15              ;MR1 IN ERROR READING GAP
3619  012656  104016                      ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3620  012660  104032                      ERROR   32              ;MR1 IN ERROR READING SYNC
3621  012662  104033                      ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3622
3623  012664  012737  000001  002346      MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
3624  012672  012737  004066  002350      MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3625  012700  004437  030774              JSR     R4,DREAD        ;GO SIMULATE DATA READ
3626  012704  000400                      400                     ;NUMBER OF WORDS TO SIMULATE
3627  012706  104034                      ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3628  012710  104035                      ERROR   35              ;ECC ERROR READING DATA
3629  012712  104036                      ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3630  012714  104041                      ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3631  012716  104042                      ERROR   42              ;ERR IN ECC PAT CALC.
3632  012720  104043                      ERROR   43              ;ERR IN ECC POS COUNT
3633
3634
3635  012722  012737  000020  002326      MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3636  012730  012700  000005              MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
3637  012734  012701  013672              MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3638  012740  005037  001214              CLR     $TMP5           ;CLEAR SWITCH
3639
3640  012744  004737  033414      3$:     JSR     PC,RDBIT        ;GO READ BIT
3641  012750  004737  032260              JSR     PC,ECCGEN       ;GENERATE ECC
3642  012754  016237  000032  002214      MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
3643  012762  016237  000030  002212      MOV     RKECPS(R2),T.ECPS       ;GET POSITION
3644  012770  005237  002350              INC     ECPOSX          ;BUMP POSITION EXPECTED
3645  012774  013737  002350  002252      MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
3646  013002  023737  002254  002214      CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
3647  013010  001401                      BEQ     1$              ;YES - SKIP
3648  013012  104042                      ERROR   42
3649  013014  023737  002252  002212  1$: CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
3650  013022  001401                      BEQ     2$              ;YES - SKIP
3651  013024  104043                      ERROR   43
3652  013026  005237  002326      2$:     INC     BITCNT          ;BUMP BIT COUNT
3653  013032  005301                      DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
3654  013034  001343                      BNE     3$              ;NOT YET ZERO - LOOP
3655  013036  005300                      DEC     R0              ;DEC PASS THRU 0 COUNT
3656  013040  001406                      BEQ     4$              ;IF 0 - SKIP
3657  013042  012737  177777  002350      MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
3658  013050  012701  020000              MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
3659  013054  000733                      BR      3$              ;GO DO LOOP AGAIN
3660
3661  013056  005737  001214      4$:     TST     $TMP5           ;TEST SWITCH
3662  013062  001012                      BNE     5$              ;ALREADY SET - SKIP
3663  013064  012701  005276              MOV     #5276,R1        ;SET COUNT TO FINAL POSITION
3664  013070  012700  000001              MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
3665  013074  012737  177777  002350      MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
3666  013102  005237  001214              INC     $TMP5           ;BUMP SWITCH
3667  013106  000716                      BR      3$              ;GO DO LOOP AGAIN
3668
3669  013110  004737  033414      5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
3670  013114  004437  034154              JSR     R4,GETREG       ;GET '611 REGISTERS
3671  013120  013737  002260  002220      MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
3672  013126  052737  100200  002220      BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
```

```
3673  013134  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
3674  013142  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3675  013150  001402                          BEQ     6$              ;YES - SKIP
3676  013152  104044                          ERROR   44
3677  013154  000410                          BR      7$
3678  013156  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
3679  013164  001404                          BEQ     7$              ;NO SKIP
3680  013166  012737  100000  002234          MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
3681  013174  104045                          ERROR   45
3682
3683  013176                          7$:
3684
3685                                  ;;**************************************************************
3686                                  ;*TEST 21       READ DATA WITH 11 BIT BURST ERROR
3687                                  ;*
3688                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3689                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3690                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3691                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3692                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
3693                                  ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
3694                                  ;*      AN 11 BIT BURST ERROR.  VERIFY THE COUNING OF THE POSITION
3695                                  ;*      REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
3696                                  ;*      MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
3697                                  ;*
3698                                  ;;**************************************************************
3699  013176  000004                  TST21:  SCOPE
3700  013200  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
3701
3702  013206  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3703
3704  013212  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3705
3706  013220  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
3707  013224  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3708  013230  000000                          0                       ;CYLINDER 0
3709  013232     000                          .BYTE   0               ;SECTOR 0
3710  013233     000                          .BYTE   0               ;TRACK 0
3711  013234  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3712  013236  177400                          -400                    ;WORD COUNT -400
3713  013240  000021                          RDDATA                  ;COMMAND RDDATA
3714
3715  013242  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3716
3717  013246  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
3718  013252  000005                          5                       ;PATTERN 5
3719
3720  013254  012700  000530                  MOV     #530,R0         ;SET INDEX INTO BUFFER
3721
3722  013260  012737  126073  052444          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
3723  013266  012737  151052  052446          MOV     #151052,OBUFF+1002
3724  013274  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
3725  013302  042760  005000  051444          BIC     #BIT9!BIT11,OBUFF(R0)   ;CREATE 11 BIT BURST ERROR
3726  013310  052760  050100  051444          BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
3727  013316  062700  000002                  ADD     #2,R0           ;BUMP TO NEXT WORD
3728  013322  052760  000001  051444          BIS     #BIT0,OBUFF(R0)
```

```
3729
3730  013330  004437  034072              JSR     R4,OPSTRT       ;START THE OPERATION
3731  013334  004437  025100              JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3732  013340  104004                      ERROR   4               ;CS1 MISCOMPARE
3733  013342  104005                      ERROR   5               ;MR1     "
3734  013344  104006                      ERROR   6               ;MR2     "
3735  013346  104007                      ERROR   7               ;MR3     "
3736
3737  013350  004437  025062              JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3738
3739  013354  004437  025504              JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3740  013360  104010                      ERROR   10              ;MR1 CONTENTS IN ERROR
3741  013362  104011                      ERROR   11              ;CS1 CONTENTS IN ERROR
3742
3743  013364  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3744  013370  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
3745  013372  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3746  013374  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3747
3748  013376  004437  030276              JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3749  013402  104015                      ERROR   15              ;MR1 IN ERROR READING GAP
3750  013404  104016                      ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3751  013406  104032                      ERROR   32              ;MR1 IN ERROR READING SYNC
3752  013410  104033                      ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3753
3754  013412  012737  000001  002346      MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
3755  013420  012737  004066  002350      MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3756  013426  004437  030774              JSR     R4,DREAD        ;GO SIMULATE DATA READ
3757  013432  000400                      400                     ;NUMBER OF WORDS TO SIMULATE
3758  013434  104034                      ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3759  013436  104035                      ERROR   35              ;ECC ERROR READING DATA
3760  013440  104036                      ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3761  013442  104041                      ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3762  013444  104042                      ERROR   42              ;ERR IN ECC PAT CALC.
3763  013446  104043                      ERROR   43              ;ERR IN ECC POS COUNT
3764
3765
3766  013450  012737  000020  002326      MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3767  013456  012700  000005              MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
3768  013462  012701  013672              MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3769  013466  005037  001214              CLR     $TMP5           ;CLEAR SWITCH
3770
3771  013472  004737  033414      3$:     JSR     PC,RDBIT        ;GO READ BIT
3772  013476  004737  032260              JSR     PC,ECCGEN       ;GENERATE ECC
3773  013502  016237  000032  002214      MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
3774  013510  016237  000030  002212      MOV     RKECPS(R2),T.ECPS       ;GET POSITION
3775  013516  005237  002350              INC     ECPOSX          ;BUMP POSITION EXPECTED
3776  013522  013737  002350  002252      MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
3777  013530  023737  002254  002214      CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
3778  013536  001401                      BEQ     1$              ;YES - SKIP
3779  013540  104042                      ERROR   42
3780  013542  023737  002252  002212  1$: CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
3781  013550  001401                      BEQ     2$              ;YES - SKIP
3782  013552  104043                      ERROR   43
3783  013554  005237  002326      2$:     INC     BITCNT          ;BUMP BIT COUNT
3784  013560  005301                      DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
```

```
3785  013562  001343                        BNE     3$              ;NOT YET ZERO - LOOP
3786  013564  005300                        DEC     R0              ;DEC PASS THRU 0 COUNT
3787  013566  001406                        BEQ     4$              ;IF 0 - SKIP
3788  013570  012737  177777  002350        MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
3789  013576  012701  020000                MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
3790  013602  000733                        BR      3$              ;GO DO LOOP AGAIN
3791
3792  013604  005737  001214          4$:   TST     $TMP5           ;TEST SWITCH
3793  013610  001012                        BNE     5$              ;ALREADY SET - SKIP
3794  013612  012701  005310                MOV     #5310,R1        ;SET COUNT TO FINAL POSITION
3795  013616  012700  000001                MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
3796  013622  012737  177777  002350        MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
3797  013630  005237  001214                INC     $TMP5           ;BUMP SWITCH
3798  013634  000716                        BR      3$              ;GO DO LOOP AGAIN
3799
3800  013636  004737  033414          5$:   JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
3801  013642  004437  034154                JSR     R4,GETREG       ;GET '611 REGISTERS
3802  013646  013737  002260  002220        MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
3803  013654  052737  100200  002220        BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
3804  013662  042737  000001  002220        BIC     #GO,E.CS1       ;CLEAR GO
3805  013670  023737  002160  002220        CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3806  013676  001402                        BEQ     6$              ;YES - SKIP
3807  013700  104044                        ERROR   44
3808  013702  000410                        BR      7$
3809  013704  032737  000100  002174  6$:   BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
3810  013712  001404                        BEQ     7$              ;NO SKIP
3811  013714  012737  100000  002234        MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
3812  013722  104045                        ERROR   45
3813
3814  013724                          7$:
3815
3816                                        ;;*****************************************************************
3817                                        ;*TEST 22       READ DATA WITH 12 BIT BURST ERROR
3818                                        ;*
3819                                        ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3820                                        ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3821                                        ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
3822                                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
3823                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
3824                                        ;*      GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
3825                                        ;*      A 12 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
3826                                        ;*      POSITION REGISTER.  MAKE SURE DATA CHECK, ECC HARD,
3827                                        ;*      AND CONTROLLER ERROR SETS.
3828                                        ;*
3829                                        ;;*****************************************************************
3830  013724  000004                TST22:  SCOPE
3831  013726  012737  000001  001234        MOV     #1,$TIMES       ;;DO 1 ITERATION
3832
3833  013734  005037  002352                CLR     BSEDES          ;CLEAR BAD SEC DESIRED
3834  013740  012762  100000  000000        MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
3835
3836  013746  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
3837  013752  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
3838  013756  000000                        0                       ;CYLINDER 0
3839  013760     000                        .BYTE   0               ;SECTOR 0
3840  013761     000                        .BYTE   0               ;TRACK 0
```

```
3841  013762  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
3842  013764  177400                          -400                    ;WORD COUNT -400
3843  013766  000021                          RDDATA                  ;COMMAND RDDATA
3844
3845  013770  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
3846
3847  013774  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
3848  014000  000005                          5                       ;PATTERN 5
3849
3850  014002  012700  000530                  MOV     #530,R0         ;SET INDEX INTO BUFFER
3851
3852  014006  012737  126073  052444          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
3853  014014  012737  151052  052446          MOV     #151052,OBUFF+1002
3854  014022  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
3855  014030  042760  005000  051444          BIC     #BIT9!BIT11,OBUFF(R0)   ;CREATE 11 BIT BURST ERROR
3856  014036  052760  050100  051444          BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
3857  014044  062700  000002                  ADD     #2,R0           ;BUMP TO NEXT WORD
3858  014050  052760  000001  051444          BIS     #BIT0,OBUFF(R0)
3859  014056  042760  000002  051444          BIC     #BIT1,OBUFF(R0) ;MAKE IT A 12 BIT BURST ERROR
3860
3861  014064  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
3862  014070  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
3863  014074  104004                          ERROR   4               ;CS1 MISCOMPARE
3864  014076  104005                          ERROR   5               ;MR1     "
3865  014100  104006                          ERROR   6               ;MR2     "
3866  014102  104007                          ERROR   7               ;MR3     "
3867
3868  014104  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
3869
3870  014110  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
3871  014114  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
3872  014116  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
3873
3874  014120  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
3875  014124  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
3876  014126  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
3877  014130  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
3878
3879  014132  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
3880  014136  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
3881  014140  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
3882  014142  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
3883  014144  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
3884
3885  014146  012737  000001  002346          MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
3886  014154  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
3887  014162  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
3888  014166  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
3889  014170  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
3890  014172  104035                          ERROR   35              ;ECC ERROR READING DATA
3891  014174  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
3892  014176  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
3893  014200  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
3894  014202  104043                          ERROR   43              ;ERR IN ECC POS COUNT
3895
3896
```

```
3897  014204  012737  000020  002326          MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
3898  014212  012700  000005                  MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
3899  014216  012701  013672                  MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
3900  014222  005037  001214                  CLR     $TMP5           ;CLEAR SWITCH
3901
3902  014226  004737  033414          3$:     JSR     PC,RDBIT        ;GO READ BIT
3903  014232  004737  032260                  JSR     PC,ECCGEN       ;GENERATE ECC
3904  014236  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
3905  014244  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
3906  014252  005237  002350                  INC     ECPOSX          ;BUMP POSITION EXPECTED
3907  014256  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
3908  014264  023737  002254  002214          CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
3909  014272  001401                          BEQ     1$              ;YES - SKIP
3910  014274  104042                          ERROR   42
3911  014276  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
3912  014304  001401                          BEQ     2$              ;YES - SKIP
3913  014306  104043                          ERROR   43
3914  014310  005237  002326          2$:     INC     BITCNT          ;BUMP BIT COUNT
3915  014314  005301                          DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
3916  014316  001343                          BNE     3$              ;NOT YET ZERO - LOOP
3917  014320  005300                          DEC     R0              ;DEC PASS THRU 0 COUNT
3918  014322  001406                          BEQ     4$              ;IF 0 - SKIP
3919  014324  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
3920  014332  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
3921  014336  000733                          BR      3$              ;GO DO LOOP AGAIN
3922
3923  014340  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
3924  014344  001012                          BNE     5$              ;ALREADY SET - SKIP
3925  014346  012701  010041                  MOV     #10041,R1       ;SET COUNT TO FINAL POSITION
3926  014352  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
3927  014356  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
3928  014364  005237  001214                  INC     $TMP5           ;BUMP SWITCH
3929  014370  000716                          BR      3$              ;GO DO LOOP AGAIN
3930
3931  014372  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
3932  014376  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
3933  014402  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
3934  014410  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
3935  014416  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
3936  014424  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
3937  014432  001402                          BEQ     6$              ;YES - SKIP
3938  014434  104044                          ERROR   44
3939  014436  000410                          BR      7$
3940  014440  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
3941  014446  001004                          BNE     7$              ;YES -  SKIP
3942  014450  012737  100100  002234          MOV     #DCK!ECH,E.ER   ;SET EXPECTED ERROR REG
3943  014456  104045                          ERROR   45
3944
3945  014460                          7$:
3946
3947                                  ;;*****************************************************************
3948                                  ;*TEST 23       READ DATA WITH 23 BIT BURST ERROR
3949                                  ;*
3950                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
3951                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
3952                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
```

```
                                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
                                        ;*      GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
                                        ;*      A 23 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
                                        ;*      POSITION REGISTER.  MAKE SURE DATA CHECK, ECC HARD,
                                        ;*      AND CONTROLLER ERROR SETS.
                                        ;*
                                        ;;*********************************************************************
014460  000004                  TST23:  SCOPE
014462  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION

014470  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
014474  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

014502  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
014506  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
014512  000000                          0                       ;CYLINDER 0
014514     000                          .BYTE   0               ;SECTOR 0
014515     000                          .BYTE   0               ;TRACK 0
014516  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
014520  177400                          -400                    ;WORD COUNT -400
014522  000021                          RDDATA                  ;COMMAND RDDATA

014524  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER

014530  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
014534  000005                          5                       ;PATTERN 5

014536  012700  000530                  MOV     #530,R0         ;SET INDEX INTO BUFFER

014542  012737  126073  052444          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
014550  012737  151052  052446          MOV     #151052,OBUFF+1002
014556  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
014564  042760  005000  051444          BIC     #BIT9!BIT11,OBUFF(R0)   ;CREATE 11 BIT BURST ERROR
014572  052760  050100  051444          BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
014600  062700  000002                  ADD     #2,R0           ;BUMP TO NEXT WORD
014604  052760  000001  051444          BIS     #BIT0,OBUFF(R0)
014612  052760  010000  051444          BIS     #BIT12,OBUFF(R0) ;MAKE IT A 23 BIT BURST ERROR

014620  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
014624  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
014630  104004                          ERROR   4               ;CS1 MISCOMPARE
014632  104005                          ERROR   5               ;MR1     "
014634  104006                          ERROR   6               ;MR2     "
014636  104007                          ERROR   7               ;MR3     "

014640  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

014644  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
014650  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
014652  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR

014654  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
014660  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
014662  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
014664  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
```

```
                                                        ;GO READ GAP AND SYNC
014666  004437  030276          JSR     R4,DRGPSN
014672  104015                  ERROR   15              ;MR1 IN ERROR READING GAP
014674  104016                  ERROR   16              ;CS1 IN ERROR AFTER READING GAP
014676  104032                  ERROR   32              ;MR1 IN ERROR READING SYNC
014700  104033                  ERROR   33              ;CS1 IN ERROR AFTER READING SYNC

014702  012737  000001  002346  MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
014710  012737  004066  002350  MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
014716  004437  030774          JSR     R4,DREAD        ;GO SIMULATE DATA READ
014722  000400                  400                     ;NUMBER OF WORDS TO SIMULATE
014724  104034                  ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
014726  104035                  ERROR   35              ;ECC ERROR READING DATA
014730  104036                  ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
014732  104041                  ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
014734  104042                  ERROR   42              ;ERR IN ECC PAT CALC.
014736  104043                  ERROR   43              ;ERR IN ECC POS COUNT

014740  012737  000020  002326  MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
014746  012700  000005          MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
014752  012701  013672          MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
014756  005037  001214          CLR     $TMP5           ;CLEAR SWITCH

014762  004737  033414  3$:     JSR     PC,RDBIT        ;GO READ BIT
014766  004737  032260          JSR     PC,ECCGEN       ;GENERATE ECC
014772  016237  000032  002214  MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
015000  016237  000030  002212  MOV     RKECPS(R2),T.ECPS       ;GET POSITION
015006  005237  002350          INC     ECPOSX          ;BUMP POSITION EXPECTED
015012  013737  002350  002252  MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
015020  023737  002254  002214  CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
015026  001401                  BEQ     1$              ;YES - SKIP
015030  104042                  ERROR   42
015032  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
015040  001401                  BEQ     2$              ;YES - SKIP
015042  104043                  ERROR   43
015044  005237  002326  2$:     INC     BITCNT          ;BUMP BIT COUNT
015050  005301                  DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
015052  001343                  BNE     3$              ;NOT YET ZERO - LOOP
015054  005300                  DEC     R0              ;DEC PASS THRU 0 COUNT
015056  001406                  BEQ     4$              ;IF 0 - SKIP
015060  012737  177777  002350  MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
015066  012701  020000          MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
015072  000733                  BR      3$              ;GO DO LOOP AGAIN

015074  005737  001214  4$:     TST     $TMP5           ;TEST SWITCH
015100  001012                  BNE     5$              ;ALREADY SET - SKIP
015102  012701  010041          MOV     #10041,R1       ;SET COUNT TO FINAL POSITION
015106  012700  000001          MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
015112  012737  177777  002350  MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
015120  005237  001214          INC     $TMP5           ;BUMP SWITCH
015124  000716                  BR      3$              ;GO DO LOOP AGAIN

015126  004737  033414  5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
015132  004437  034154          JSR     R4,GETREG       ;GET '611 REGISTERS
015136  013737  002260  002220  MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
```

```
4065  015144  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
4066  015152  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
4067  015160  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4068  015166  001402                          BEQ     6$              ;YES - SKIP
4069  015170  104044                          ERROR   44
4070  015172  000410                          BR      7$
4071  015174  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
4072  015202  001004                          BNE     7$              ;YES - SKIP
4073  015204  012737  100100  002234          MOV     #DCK!ECH,E.ER   ;SET EXPECTED ERROR REG
4074  015212  104045                          ERROR   45
4075
4076  015214                          7$:
4077
4078                                  ;;*****************************************************************
4079                                  ;*TEST 24       READ DATA WITH 1 BIT ECC WORD ERROR
4080                                  ;*
4081                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4082                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4083                                  ;*      OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4084                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4085                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE, A
4086                                  ;*      GOOD HEADER, 400 DATA WORDS AND AN ECC INDICATING
4087                                  ;*      A 1 BIT ERROR IN ECC WORD 2, BIT 6.  VERIFY THE
4088                                  ;*      COUNTING OF THE POSITION REGISTER.  MAKE SURE DATA
4089                                  ;*      CHECK AND CONTROLLER ERROR SETS.
4090                                  ;*
4091                                  ;;*****************************************************************
4092  015214  000004          ^ST24:  SCOPE
4093  015216  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
4094
4095  015224  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
4096  015230  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4097
4098  015236  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
4099  015242  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4100  015246  000000                          0                       ;CYLINDER 0
4101  015250      000                         .BYTE   0               ;SECTOR 0
4102  015251      000                         .BYTE   0               ;TRACK 0
4103  015252  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
4104  015254  177400                          -400                    ;WORD COUNT -400
4105  015256  000021                          RDDATA                  ;COMMAND RDDATA
4106
4107  015260  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4108
4109  015264  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
4110  015270  000005                          5                       ;PATTERN 5
4111
4112
4113  015272  012700  001002                  MOV     #1002,R0        ;SET INDEX INTO BUFFER
4114  015276  012737  126073  052444          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
4115  015304  012737  151052  052446          MOV     #151052,OBUFF+1002
4116  015312  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
4117  015320  032760  000100  051444          BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
4118  015326  001004                          BNE     100$            ;YES - SKIP
4119  015330  052760  000100  051444          BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
4120  015336  000403                          BR      101$
```

```
4121  015340  042760  000100  051444  100$:   BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
4122  015346                          101$:
4123
4124  015346  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
4125  015352  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4126  015356  104004                          ERROR   4               ;CS1 MISCOMPARE
4127  015360  104005                          ERROR   5               ;MR1     "
4128  015362  104006                          ERROR   6               ;MR2     "
4129  015364  104007                          ERROR   7               ;MR3     "
4130
4131  015366  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4132
4133  015372  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4134  015376  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
4135  015400  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
4136
4137  015402  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4138  015406  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
4139  015410  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4140  015412  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4141
4142  015414  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4143  015420  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
4144  015422  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4145  015424  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
4146  015426  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4147
4148  015430  012737  000001  002346          MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
4149  015436  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
4150  015444  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
4151  015450  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
4152  015452  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4153  015454  104035                          ERROR   35              ;ECC ERROR READING DATA
4154  015456  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4155  015460  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4156  015462  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
4157  015464  104043                          ERROR   43              ;ERR IN ECC POS COUNT
4158
4159
4160  015466  012737  000020  002326          MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
4161  015474  012700  000005                  MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
4162  015500  012701  013672                  MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
4163  015504  005037  001214                  CLR     $TMP5           ;CLEAR SWITCH
4164
4165  015510  004737  033414          3$:     JSR     PC,RDBIT        ;GO READ BIT
4166  015514  004737  032260                  JSR     PC,ECCGEN       ;GENERATE ECC
4167  015520  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
4168  015526  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
4169  015534  005237  002350                  INC     ECPOSX          ;BUMP POSITION EXPECTED
4170  015540  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
4171  015546  023737  002254  002214          CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
4172  015554  001401                          BEQ     1$              ;YES - SKIP
4173  015556  104042                          ERROR   42
4174  015560  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
4175  015566  001401                          BEQ     2$              ;YES - SKIP
4176  015570  104043                          ERROR   43
```

```
4177  015572  005237  002326          2$:     INC     BITCNT          ;BUMP BIT COUNT
4178  015576  005301                          DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
4179  015600  001343                          BNE     3$              ;NOT YET ZERO - LOOP
4180  015602  005300                          DEC     R0              ;DEC PASS THRU 0 COUNT
4181  015604  001406                          BEQ     4$              ;IF 0 - SKIP
4182  015606  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
4183  015614  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
4184  015620  000733                          BR      3$              ;GO DO LOOP AGAIN
4185
4186  015622  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
4187  015626  001012                          BNE     5$              ;ALREADY SET - SKIP
4188  015630  012701  010016                  MOV     #10016,R1       ;SET COUNT TO FINAL POSITION
4189  015634  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
4190  015640  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
4191  015646  005237  001214                  INC     $TMP5           ;BUMP SWITCH
4192  015652  000716                          BR      3$              ;GO DO LOOP AGAIN
4193
4194  015654  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
4195  015660  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
4196  015664  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
4197  015672  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
4198  015700  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
4199  015706  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4200  015714  001402                          BEQ     6$              ;YES - SKIP
4201  015716  104044                          ERROR   44
4202  015720  000410                          BR      7$
4203  015722  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
4204  015730  001404                          BEQ     7$              ;YES - SKIP
4205  015732  012737  100000  002234          MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
4206  015740  104045                          ERROR   45
4207
4208  015742                          7$:
4209
4210                                  ;;*****************************************************************
4211                                  ;*TEST 25       18 BIT NPR WRITING OF MEMORY
4212                                  ;*
4213                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4214                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4215                                  ;*      OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4216                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4217                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4218                                  ;*      A GOOD HEADER, 40 DATA WORDS.  VERIFY THAT THE WORDS
4219                                  ;*      ARE WRITTEN PROPERLY IN MEMORY.
4220                                  ;*
4221                                  ;;*****************************************************************
4222  015742  000004                  TST25:  SCOPE
4223  015744  012737  000012  001234          MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
4224
4225  015752  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
4226  015756  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4227
4228  015764  005037  002344                  CLR     ECCSRC          ;CLEAR ECC SOURCE
4229  015770  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
4230  015774  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4231  016000  000000                          0                       ;CYLINDER 0
4232  016002     000                          .BYTE   0               ;SECTOR 0
```

```
4233  016003  000                       .BYTE   0               ;TRACK 0
4234  016004  050440                    IBUFF                   ;BUFFER ADDRESS IBUFF
4235  016006  177400                    -400                    ;WORD COUNT -400
4236  016010  010021                    RDDATA!CFMT                     ;COMMAND RDDATA!CFMT
4237
4238  016012  004437  033566            JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4239
4240  016016  004437  033612            JSR     R4,BLDDAT       ;GO BUILD DATA
4241  016022  000003                    3                       ;PATTERN 3
4242
4243
4244  016024  004437  034072            JSR     R4,OPSTRT       ;START THE OPERATION
4245  016030  004437  025100            JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4246  016034  104004                    ERROR   4               ;CS1 MISCOMPARE
4247  016036  104005                    ERROR   5               ;MR1     "
4248  016040  104006                    ERROR   6               ;MR2     "
4249  016042  104007                    ERROR   7               ;MR3     "
4250
4251  016044  004437  025062            JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4252
4253  016050  004437  025504            JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4254  016054  104010                    ERROR   10              ;MR1 CONTENTS IN ERROR
4255  016056  104011                    ERROR   11              ;CS1 CONTENTS IN ERROR
4256
4257  016060  004437  026042            JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4258  016064  104012                    ERROR   12              ;MR1 CONTENTS IN ERROR
4259  016066  104013                    ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4260  016070  104014                    ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4261
4262  016072  004437  030276            JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4263  016076  104015                    ERROR   15              ;MR1 IN ERROR READING GAP
4264  016100  104016                    ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4265  016102  104032                    ERROR   32              ;MR1 IN ERROR READING SYNC
4266  016104  104033                    ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4267
4268  016106  012737  000000  002346    MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
4269  016114  012737  005066  002350    MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
4270  016122  004437  030774            JSR     R4,DREAD        ;GO SIMULATE DATA READ
4271  016126  000060                    60                      ;NUMBER OF WORDS TO SIMULATE
4272  016130  104034                    ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4273  016132  104035                    ERROR   35              ;ECC ERROR READING DATA
4274  016134  104036                    ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4275  016136  104041                    ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4276  016140  104042                    ERROR   42              ;ERR IN ECC PAT CALC.
4277  016142  104043                    ERROR   43              ;ERR IN ECC POS COUNT
4278
4279
4280  016144  012700  051444            MOV     #OBUFF,R0       ;SET POINTER TO GOOD DATA
4281  016150  012701  050440            MOV     #IBUFF,R1       ;SET POINTER TO INPUT DATA
4282
4283  016154  012704  000012            MOV     #10.,R4         ;SET ERROR LIMIT COUNT
4284  016160  005037  001236            CLR     $ESCAPE         ;CLEAR ESCAPE
4285  016164  005037  001220            CLR     $TMP7           ;CLEAR ERROR REPORT SWITCH
4286  016170  012703  000040            MOV     #40,R3          ;SET COMPARE LIMIT
4287  016174  005005                    CLR     R5              ;CLEAR WORD COUNTER
4288
```

```
4289  016176  021011                1$:     CMP     (R0),(R1)       ;COMPARE DATA
4290  016200  001005                        BNE     2$              ;SKIP IF NOT EQUAL
4291  016202  005303                4$:     DEC     R3              ;ELSE DEC WORD COUNT LIMIT
4292  016204  001424                        BEQ     5$              ;IF 0 - SKIP TO EXIT
4293  016206  005205                        INC     R5              ;BUMP WORD COUNT
4294  016210  022021                        CMP     (R0)+,(R1)+     ;BUMP DATA POINTERS
4295  016212  000771                        BR      1$              ;LOOP
4296
4297  016214  005304                2$:     DEC     R4              ;DEC ERROR LIMIT
4298  016216  001417                        BEQ     5$              ;IF 0 - SKIP
4299
4300  016220  011037  001162                MOV     (R0),$REG0      ;GET GOOD WORD FOR REPORT
4301  016224  011137  001164                MOV     (R1),$REG1      ;GET BAD WORD
4302  016230  010537  001166                MOV     R5,$REG2        ;GET WORD NUMBER
4303
4304  016234  005737  001220                TST     $TMP7           ;DECIDE WHICH ERROR REPORT TO USE
4305  016240  001004                        BNE     3$              ;SKIP IF NOT 0
4306  016242  005237  001220                INC     $TMP7           ;ELSE BUMP $TMP7
4307  016246  104037                        ERROR   37              ;REPORT FIRST ERROR LINE
4308  016250  000754                        BR      4$              ;SKIP
4309
4310  016252  104040                3$:     ERROR 40                ;REPORT ALL OTHER LINES
4311  016254  000752                        BR      4$              ;SKIP
4312
4313  016256                        5$:
4314
4315                                ;;*****************************************************************
4316                                ;*TEST 26       18 BIT READ DATA WITH NO ERROR
4317                                ;*
4318                                ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4319                                ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4320                                ;*      OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4321                                ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4322                                ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4323                                ;*      A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
4324                                ;*      VERIFY THAT COMMAND COMPLETES WITHOUT ERROR.
4325                                ;*
4326                                ;;*****************************************************************
4327  016256  000004                TST26:  SCOPE
4328  016260  012737  000001  001234        MOV     #1,$TIMES       ;;DO 1 ITERATION
4329
4330  016266  005037  002352                CLR     BSEDES          ;CLEAR BAD SEC DESIRED
4331  016272  012762  100000  000000        MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4332  016300  005037  002344                CLR     ECCSRC          ;CLEAR ECC SOURCE FLAG
4333  016304  005037  002344                CLR     ECCSRC          ;CLEAR ECC SOURCE
4334  016310  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
4335  016314  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4336  016320  000000                        0                       ;CYLINDER 0
4337  016322     000                        .BYTE   0               ;SECTOR 0
4338  016323     000                        .BYTE   0               ;TRACK 0
4339  016324  050440                        IBUFF                   ;BUFFER ADDRESS IBUFF
4340  016326  177400                        -400                    ;WORD COUNT -400
4341  016330  010021                        RDDATA!CFMT                     ;COMMAND RDDATA!CFMT
4342
4343  016332  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4344
```

```
4345  016336  004437  033612              JSR     R4,BLDDAT       ;GO BUILD DATA
4346  016342  000004                      4                       ;PATTERN 4
4347
4348
4349  016344  004437  034072              JSR     R4,OPSTRT       ;START THE OPERATION
4350  016350  004437  025100              JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4351  016354  104004                      ERROR   4               ;CS1 MISCOMPARE
4352  016356  104005                      ERROR   5               ;MR1     "
4353  016360  104006                      ERROR   6               ;MR2     "
4354  016362  104007                      ERROR   7               ;MR3     "
4355
4356  016364  004437  025062              JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4357
4358  016370  004437  025504              JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4359  016374  104010                      ERROR   10              ;MR1 CONTENTS IN ERROR
4360  016376  104011                      ERROR   11              ;CS1 CONTENTS IN ERROR
4361
4362  016400  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4363  016404  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
4364  016406  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4365  016410  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4366
4367  016412  004437  030276              JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4368  016416  104015                      ERROR   15              ;MR1 IN ERROR READING GAP
4369  016420  104016                      ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4370  016422  104032                      ERROR   32              ;MR1 IN ERROR READING SYNC
4371  016424  104033                      ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4372
4373  016426  012737  000000  002346      MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
4374  016434  012737  005066  002350      MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
4375  016442  004437  030774              JSR     R4,DREAD        ;GO SIMULATE DATA READ
4376  016446  000400                      400                     ;NUMBER OF WORDS TO SIMULATE
4377  016450  104034                      ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4378  016452  104035                      ERROR   35              ;ECC ERROR READING DATA
4379  016454  104036                      ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4380  016456  104041                      ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4381  016460  104042                      ERROR   42              ;ERR IN ECC PAT CALC.
4382  016462  104043                      ERROR   43              ;ERR IN ECC POS COUNT
4383
4384
4385  016464  012700  003100              MOV     #40.*40.,R0     ;SET COUNT TO EMPTY SILO
4386
4387  016470  012762  000440  000026  1$: MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
4388  016476  012762  000040  000026      MOV     #DMD,RKMR1(R2)
4389  016504  005300                      DEC     R0              ;DEC COUNT
4390  016506  001370                      BNE     1$              ;LOOP UNTIL 0
4391
4392  016510  004437  034154              JSR     R4,GETREG       ;GET 611 REGS
4393  016514  013737  002260  002220      MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
4394  016522  042737  000001  002220      BIC     #GO,E.CS1       ;CLEAR GO BIT
4395  016530  052737  000200  002220      BIS     #RDY,E.CS1      ;SET READY BIT
4396  016536  023737  002160  002220      CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4397  016544  001401                      BEQ     2$              ;YES - SKIP
4398  016546  104056                      ERROR   56              ;"CS1 IN ERROR AFTER READ DATA"
4399  016550                          2$:
4400                                  ;;**************************************************************
```

```
                                        ;*TEST 27       18 BIT READ DATA WITH DCK
                                        ;*
                                        ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                        ;*      OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        ;*      A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                                        ;*      A ONE BIT BURST ERROR BEYOND THE 4128TH BIT OF THE SECTOR.
                                        ;*      VERIFY THE COUNTING OF THE POSITION REGISTER AND THE
                                        ;*      FINAL PATTERN AND POSITION.  MAKE SURE DATA CHECK AND
                                        ;*      CONTROLLER ERROR SETS.
                                        ;*
                                        ;**************************************************************
016550  000004                  TST27:  SCOPE
016552  012737  000001  001234        MOV     #1,$TIMES       ;;DO 1 ITERATION

016560  005037  002352                CLR     BSEDES          ;CLEAR BAD SEC DESIRED
016564  012762  100000  000000        MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

016572  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
016576  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
016602  000000                        0                       ;CYLINDER 0
016604     000                        .BYTE   0               ;SECTOR 0
016605     000                        .BYTE   0               ;TRACK 0
016606  050440                        IBUFF                   ;BUFFER ADDRESS IBUFF
016610  177400                        -400                    ;WORD COUNT -400
016612  010021                        RDDATA!CFMT                     ;COMMAND RDDATA!CFMT

016614  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER

016620  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
016624  000007                        7                       ;PATTERN 7


016626  012700  000734                MOV     #734,R0         ;SET INDEX INTO BUFFER
016632  012737  006030  052444        MOV     #6030,OBUFF+1000        ;STORE ECC WORDS
016640  012737  045070  052446        MOV     #45070,OBUFF+1002
016646  012737  000001  002344        MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
016654  032760  000100  051444        BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
016662  001004                        BNE     100$            ;YES - SKIP
016664  052760  000100  051444        BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
016672  000403                        BR      101$
016674  042760  000100  051444  100$:BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
016702                        101$:

016702  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
016706  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
016712  104004                        ERROR   4               ;CS1 MISCOMPARE
016714  104005                        ERROR   5               ;MR1     "
016716  104006                        ERROR   6               ;MR2     "
016720  104007                        ERROR   7               ;MR3     "

016722  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

016726  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
```

```
4457  016732  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
4458  016734  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
4459
4460  016736  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4461  016742  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
4462  016744  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4463  016746  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4464
4465  016750  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4466  016754  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
4467  016756  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4468  016760  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
4469  016762  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4470
4471  016764  012737  000001  002346          MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
4472  016772  012737  005066  002350          MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
4473  017000  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
4474  017004  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
4475  017006  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4476  017010  104035                          ERROR   35              ;ECC ERROR READING DATA
4477  017012  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4478  017014  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4479  017016  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
4480  017020  104043                          ERROR   43              ;ERR IN ECC POS COUNT
4481
4482
4483  017022  012737  000022  002326          MOV     #22,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
4484  017030  012700  000005                  MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
4485  017034  012701  012672                  MOV     #12672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
4486  017040  005037  001214                  CLR     $TMP5           ;CLEAR SWITCH
4487
4488  017044  004737  033414          3$:     JSR     PC,RDBIT        ;GO READ BIT
4489  017050  004737  032260                  JSR     PC,ECCGEN       ;GENERATE ECC
4490  017054  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
4491  017062  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
4492  017070  005237  002350                  INC     ECPOSX          ;BUMP POSITION EXPECTED
4493  017074  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
4494  017102  023737  002254  002214          CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
4495  017110  001401                          BEQ     1$              ;YES - SKIP
4496  017112  104042                          ERROR   42
4497  017114  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
4498  017122  001401                          BEQ     2$              ;YES - SKIP
4499  017124  104043                          ERROR   43
4500  017126  005237  002326          2$:     INC     BITCNT          ;BUMP BIT COUNT
4501  017132  005301                          DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
4502  017134  001343                          BNE     3$              ;NOT YET ZERO - LOOP
4503  017136  005300                          DEC     R0              ;DEC PASS THRU 0 COUNT
4504  017140  001406                          BEQ     4$              ;IF 0 - SKIP
4505  017142  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
4506  017150  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
4507  017154  000733                          BR      3$              ;GO DO LOOP AGAIN
4508
4509  017156  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
4510  017162  001012                          BNE     5$              ;ALREADY SET - SKIP
4511  017164  012701  010272                  MOV     #10272,R1       ;SET COUNT TO FINAL POSITION
4512  017170  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
```

```
4513  017174  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
4514  017202  005237  001214                  INC     $TMP5           ;BUMP SWITCH
4515  017206  000716                          BR      3$              ;GO DO LOOP AGAIN
4516
4517  017210  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
4518  017214  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
4519  017220  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
4520  017226  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
4521  017234  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
4522  017242  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4523  017250  001402                          BEQ     6$              ;YES - SKIP
4524  017252  104044                          ERROR   44
4525  017254  000410                          BR      7$
4526  017256  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
4527  017264  001404                          BEQ     7$              ;NO SKIP
4528  017266  012737  100000  002234          MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
4529  017274  104045                          ERROR   45
4530
4531  017276                          7$:
4532
4533                                  ;;**************************************************************
4534                                  ;*TEST 30       18 BIT READ DATA WITH ECH
4535                                  ;*
4536                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4537                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4538                                  ;*      OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
4539                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4540                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4541                                  ;*      A GOOD HEADER, 400 DATA WORDS AND AND ECC INDICATING
4542                                  ;*      A 12 BIT BURST ERROR.  VERIFY THE COUNTING OF THE
4543                                  ;*      POSITION REGISTER.  MAKE SURE DATE CHECK, ECC HARD,
4544                                  ;*      AND CONTROLLER ERROR SETS.
4545                                  ;*
4546                                  ;;**************************************************************
4547  017276  000004          TST30:  SCOPE
4548  017300  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
4549
4550  017306  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
4551  017312  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4552
4553  017320  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
4554  017324  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4555  017330  000000                          0                       ;CYLINDER 0
4556  017332     000                          .BYTE   0               ;SECTOR 0
4557  017333     000                          .BYTE   0               ;TRACK 0
4558  017334  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
4559  017336  177400                          -400                    ;WORD COUNT -400
4560  017340  010021                          RDDATA!CFMT                     ;COMMAND RDDATA!CFMT
4561
4562  017342  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4563
4564  017346  004437  033612                  JSR     R4,BLDDAT       ;GO BUILD DATA
4565  017352  000005                          5                       ;PATTERN 5
4566
4567
4568  017354  012700  000734                  MOV     #734,R0         ;SET INDEX INTO BUFFER
```

```
4569  017360  012737  004020  052444          MOV     #4020,OBUFF+1000        ;STORE ECC WORDS
4570  017366  012737  071720  052446          MOV     #71720,OBUFF+1002
4571  017374  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
4572  017402  042760  005000  051444          BIC     #BIT9!BIT11,OBUFF(R0)   ;CREATE 11 BIT BURST ERROR
4573  017410  052760  050100  051444          BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
4574  017416  062700  000002                  ADD     #2,R0           ;BUMP TO NEXT WORD
4575  017422  052760  000001  051444          BIS     #BIT0,OBUFF(R0)
4576  017430  042760  000002  051444          BIC     #BIT1,OBUFF(R0) ;MAKE IT A 12 BIT BURST ERROR
4577
4578  017436  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
4579  017442  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4580  017446  104004                  ERROR   4               ;CS1 MISCOMPARE
4581  017450  104005                  ERROR   5               ;MR1    "
4582  017452  104006                  ERROR   6               ;MR2    "
4583  017454  104007                  ERROR   7               ;MR3    "
4584
4585  017456  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4586
4587  017462  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4588  017466  104010                  ERROR   10              ;MR1 CONTENTS IN ERROR
4589  017470  104011                  ERROR   11              ;CS1 CONTENTS IN ERROR
4590
4591  017472  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4592  017476  104012                  ERROR   12              ;MR1 CONTENTS IN ERROR
4593  017500  104013                  ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4594  017502  104014                  ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4595
4596  017504  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4597  017510  104015                  ERROR   15              ;MR1 IN ERROR READING GAP
4598  017512  104016                  ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4599  017514  104032                  ERROR   32              ;MR1 IN ERROR READING SYNC
4600  017516  104033                  ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4601
4602  017520  012737  000001  002346          MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
4603  017526  012737  005066  002350          MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
4604  017534  004437  030774                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
4605  017540  000400                  400             ;NUMBER OF WORDS TO SIMULATE
4606  017542  104034                  ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4607  017544  104035                  ERROR   35              ;ECC ERROR READING DATA
4608  017546  104036                  ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4609  017550  104041                  ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4610  017552  104042                  ERROR   42              ;ERR IN ECC PAT CALC.
4611  017554  104043                  ERROR   43              ;ERR IN ECC POS COUNT
4612
4613
4614  017556  012737  000022  002326          MOV     #22,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
4615  017564  012700  000005                  MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
4616  017570  012701  012672                  MOV     #12672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
4617  017574  005037  001214                  CLR     $TMP5           ;CLEAR SWITCH
4618
4619  017600  004737  033414          3$:     JSR     PC,RDBIT        ;GO READ BIT
4620  017604  004737  032260                  JSR     PC,ECCGEN       ;GENERATE ECC
4621  017610  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
4622  017616  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
4623  017624  005237  002350                  INC     ECPOSX          ;BUMP POSITION EXPECTED
4624  017630  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
```

```
4625  017636  023737  002254  002214          CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
4626  017644  001401                          BEQ     1$              ;YES - SKIP
4627  017646  104042                          ERROR   42
4628  017650  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
4629  017656  001401                          BEQ     2$              ;YES - SKIP
4630  017660  104043                          ERROR   43
4631  017662  005237  002326          2$:     INC     BITCNT          ;BUMP BIT COUNT
4632  017666  005301                          DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
4633  017670  001343                          BNE     3$              ;NOT YET ZERO - LOOP
4634  017672  005300                          DEC     R0              ;DEC PASS THRU 0 COUNT
4635  017674  001406                          BEQ     4$              ;IF 0 - SKIP
4636  017676  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
4637  017704  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
4638  017710  000733                          BR      3$              ;GO DO LOOP AGAIN
4639
4640  017712  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
4641  017716  001012                          BNE     5$              ;ALREADY SET - SKIP
4642  017720  012701  011041                  MOV     #11041,R1       ;SET COUNT TO FINAL POSITION
4643  017724  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
4644  017730  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
4645  017736  005237  001214                  INC     $TMP5           ;BUMP SWITCH
4646  017742  000716                          BR      3$              ;GO DO LOOP AGAIN
4647
4648  017744  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
4649  017750  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
4650  017754  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
4651  017762  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
4652  017770  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
4653  017776  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4654  020004  001402                          BEQ     6$              ;YES - SKIP
4655  020006  104044                          ERROR   44
4656  020010  000410                          BR      7$
4657  020012  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
4658  020020  001004                          BNE     7$              ;YES -  SKIP
4659  020022  012737  100100  002234          MOV     #DCK!ECH,E.ER   ;SET EXPECTED ERROR REG
4660  020030  104045                          ERROR   45
4661
4662  020032                          7$:
4663
4664                                  .SBTTL  **SPECIAL READ TESTS
4665
4666                                  ;;*****************************************************************
4667                                  ;*TEST 31       PARTIAL SECTOR READ
4668                                  ;*
4669                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4670                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
4671                                  ;*      OF 103 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4672                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4673                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4674                                  ;*      A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
4675                                  ;*      MAKE SURE ONLY 103 WORDS GOT TRANSFERRED TO MEMORY.
4676                                  ;*
4677                                  ;;*****************************************************************
4678  020032  000004                  TST31:  SCOPE
4679  020034  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
4680
```

```
4681  020042  005037  002352                CLR     BSEDES          ;CLEAR BAD SEC DESIRED
4682  020046  012762  100000  000000        MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4683
4684  020054  005037  002344                CLR     ECCSRC          ;CLEAR ECC SOURCE
4685  020060  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
4686  020064  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4687  020070  000000                        0                       ;CYLINDER 0
4688  020072     000                        .BYTE   0               ;SECTOR 0
4689  020073     000                        .BYTE   0               ;TRACK 0
4690  020074  050440                        IBUFF                   ;BUFFER ADDRESS IBUFF
4691  020076  177675                        -103                    ;WORD COUNT -103
4692  020100  000021                        RDDATA                  ;COMMAND RDDATA
4693
4694  020102  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4695
4696  020106  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
4697  020112  000006                        6                       ;PATTERN 6
4698
4699
4700  020114  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
4701  020120  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4702  020124  104004                        ERROR   4               ;CS1 MISCOMPARE
4703  020126  104005                        ERROR   5               ;MR1     "
4704  020130  104006                        ERROR   6               ;MR2     "
4705  020132  104007                        ERROR   7               ;MR3     "
4706
4707  020134  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4708
4709  020140  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4710  020144  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
4711  020146  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
4712
4713  020150  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4714  020154  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
4715  020156  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4716  020160  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4717
4718  020162  004437  030276                JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4719  020166  104015                        ERROR   15              ;MR1 IN ERROR READING GAP
4720  020170  104016                        ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4721  020172  104032                        ERROR   32              ;MR1 IN ERROR READING SYNC
4722  020174  104033                        ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4723
4724  020176  012737  000000  002346        MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
4725  020204  012737  004066  002350        MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
4726  020212  004437  030774                JSR     R4,DREAD        ;GO SIMULATE DATA READ
4727  020216  000400                        400                     ;NUMBER OF WORDS TO SIMULATE
4728  020220  104034                        ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4729  020222  104035                        ERROR   35              ;ECC ERROR READING DATA
4730  020224  104036                        ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4731  020226  104041                        ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4732  020230  104042                        ERROR   42              ;ERR IN ECC PAT CALC.
4733  020232  104043                        ERROR   43              ;ERR IN ECC POS COUNT
4734
4735
4736  020234  013700  002164                MOV     T.BA,R0         ;GET ACTUAL BA
```

```
                                                SUB     #IBUFF,R0       ;SUB START VALUE OF BA
                                                CMP     #206,R0         ;TEST IF CORRECT NUM OF WORDS XFER
                                                BEQ     4$              ;YES - SKIP
                                                MOV     #IBUFF+206,E.BA ;SET EXPECTED BA
                                                ERROR   46              ;REPORT XFER LENGTH ERROR

                                        4$:     MOV     #3.*4.,R1       ;SET COUNT TO END OF OPERATION
                                        3$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN THE CLOCK
                                                MOV     #DMD,RKMR1(R2)
                                                DEC     R1              ;DEC THE COUNT
                                                BNE     3$              ;LOOP UNTIL ZERO

                                        1$:     JSR     R4,GETREG       ;GET 611 REGS
                                                MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
                                                BIC     #GO,E.CS1       ;CLEAR GO BIT
                                                BIS     #RDY,E.CS1      ;SET READY BIT
                                                CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
                                                BEQ     2$              ;YES - SKIP
                                                ERROR   56              ;"CS1 IN ERROR AFTER READ DATA"
                                        2$:

                                        ;;*****************************************************************
                                        ;*TEST 32       SILO UNLOAD BEFORE HEADER ERROR
                                        ;*
                                        ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                        ;*      OF 401 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        ;*      A GOOD HEADER, 400 DATA WORDS, AND A GOOD 32 BIT ECC.
                                        ;*      AFTER ECC SIMULATE A SECTOR PULSE AND A BAD SECTOR ERROR.
                                        ;*      MAKE SURE THE ENTIRE PREVIOUS SECTOR GETS LOADED INTO
                                        ;*      MEMORY BEFORE BAD SECTOR ERROR AND CONTROLLER ERROR SETS.
                                        ;*
                                        ;*****************************************************************
                                        TST32:  SCOPE
                                                MOV     #1,$TIMES       ;;DO 1 ITERATION

                                                MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

                                                CLR     BSEDES          ;CLEAR BSE DESIRED
                                                CLR     ECCSRC          ;CLEAR ECC SOURCE
                                                CLR     HIBITS          ;CLEAR HI ORDER BITS
                                                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
                                                0                       ;CYLINDER 0
                                                .BYTE   0               ;SECTOR 0
                                                .BYTE   0               ;TRACK 0
                                                IBUFF                   ;BUFFER ADDRESS IBUFF
                                                -401                    ;WORD COUNT -401
                                                RDDATA                  ;COMMAND RDDATA

                                                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER

                                                JSR     R4,BLDDAT       ;GO BUILD DATA
                                                2                       ;PATTERN 2
```

```
4793
4794
4795  020430  004437  034072              JSR     R4,OPSTRT       ;START THE OPERATION
4796  020434  004437  025100              JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4797  020440  104004                      ERROR   4               ;CS1 MISCOMPARE
4798  020442  104005                      ERROR   5               ;MR1     "
4799  020444  104006                      ERROR   6               ;MR2     "
4800  020446  104007                      ERROR   7               ;MR3     "
4801
4802  020450  004437  025062              JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4803
4804  020454  004437  025504              JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4805  020460  104010                      ERROR   10              ;MR1 CONTENTS IN ERROR
4806  020462  104011                      ERROR   11              ;CS1 CONTENTS IN ERROR
4807
4808  020464  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4809  020470  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
4810  020472  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4811  020474  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4812
4813  020476  004437  030276              JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4814  020502  104015                      ERROR   15              ;MR1 IN ERROR READING GAP
4815  020504  104016                      ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4816  020506  104032                      ERROR   32              ;MR1 IN ERROR READING SYNC
4817  020510  104033                      ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4818
4819  020512  012737  000000  002346      MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
4820  020520  012737  004066  002350      MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
4821  020526  004437  030774              JSR     R4,DREAD        ;GO SIMULATE DATA READ
4822  020532  000400                      400                     ;NUMBER OF WORDS TO SIMULATE
4823  020534  104034                      ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
4824  020536  104035                      ERROR   35              ;ECC ERROR READING DATA
4825  020540  104036                      ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
4826  020542  104041                      ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4827  020544  104042                      ERROR   42              ;ERR IN ECC PAT CALC.
4828  020546  104043                      ERROR   43              ;ERR IN ECC POS COUNT
4829
4830
4831  020550  012701  000040              MOV     #8.*4.,R1       ;SET COUNT TO END OF SECTOR
4832  020554  012762  000440  000026  6$: MOV     #DMD!MCLK,RKMR1(R2) ;RUN THE CLOCK
4833  020562  012762  000040  000026      MOV     #DMD,RKMR1(R2)
4834  020570  005301                      DEC     R1              ;DEC THE COUNT
4835  020572  001370                      BNE     6$              ;LOOP UNTIL ZERO
4836
4837  020574  052737  100000  002352      BIS     #BIT15,BSEDES   ;SET FOR BAD SECTOR ERROR
4838
4839  020602  004437  025062              JSR     R4,DSECTR       ;ISSUE SECTOR PULSE
4840
4841  020606  004437  025504              JSR     R4,DRSYNC       ;READ SYNC
4842  020612  104010                      ERROR   10              ;MR1 CONTENTS IN ERROR
4843  020614  104011                      ERROR   11              ;CS1 CONTENTS IN ERROR
4844
4845  020616  004437  026042              JSR     R4,DHDCMP       ;DO HEADER COMPARE WITH BSE ERROR
4846  020622  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
4847  020624  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4848  020626  000411                      BR      1$              ;EXPECTED RETURN (NO DA INCREMENT)
```

```
4849
4850                                    ;       IF THIS CODE IS EXECUTED, THE CONTROLLER DID NOT INHIBIT
4851                                    ;       THE ADDRESS INCREMENT BECAUSE OF THE BSE ERROR.
4852
4853  020630  005037  002352                    CLR     BSEDES          ;CLEAR BSE DESIRED SWITCH
4854  020634  013737  002274  002240            MOV     L.DCYL,E.DCYL   ;SET EXPECTED CYL
4855  020642  013737  002266  002226            MOV     L.DA,E.DA       ;SET EXPECTED SEC/TRACK
4856  020650  104047                            ERROR   47              ;UNEXPECTED ADDRESS INCREMENT
4857
4858  020652  016237  000014  002174  1$:       MOV     RKER(R2),T.ER   ;GET ERROR REG
4859  020660  032737  000200  002174            BIT     #BSE,T.ER       ;TEST IF BSE SET
4860  020666  001005                            BNE     2$              ;YES - SKIP
4861  020670  012737  000200  002234            MOV     #BSE,E.ER       ;SET EXPECTED ERROR REG
4862  020676  104051                            ERROR   51              ;"BSE DID NOT SET"
4863  020700  000453                            BR      TST33           ;;GO TO NEXT TEST
4864
4865  020702  004437  034154          2$:       JSR     R4,GETREG       ;GET 611 REGS
4866  020706  013737  002260  002220            MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
4867  020714  023737  002220  002160            CMP     E.CS1,T.CS1     ;CHECK IF CS1 OK
4868  020722  001402                            BEQ     3$              ;YES - SKIP
4869  020724  104056                            ERROR   56              ;"CS1 ERROR AFTER WRITE CHECK"
4870  020726  000440                            BR      TST33           ;;GO TO NEXT TEST
4871
4872  020730  012701  010540          3$:       MOV     #278.*16.,R1    ;SET COUNT TO NEXT ECC FIELD
4873  020734  005037  002320                    CLR     PR.BIT          ;CLEAR PRESENT AND PREVIOUS BITS
4874  020740  005037  002322                    CLR     M1.BIT
4875  020744  004737  033414          4$:       JSR     PC,RDBIT        ;SIMULATE READ BIT
4876  020750  005301                            DEC     R1              ;DEC COUNT
4877  020752  001374                            BNE     4$              ;LOOP UNTIL ZERO
4878
4879  020754  004437  034154                    JSR     R4,GETREG       ;GET 611 REGS
4880  020760  042737  000001  002220            BIC     #GO,E.CS1       ;SET UP EXPECTED CS1
4881  020766  052737  100200  002220            BIS     #RDY!CERR,E.CS1
4882  020774  023737  002220  002160            CMP     E.CS1,T.CS1     ;CHECK IF CS1 OK
4883  021002  001402                            BEQ     7$              ;YES - SKIP
4884  021004  104056                            ERROR   56              ;"CS1 ERROR AFTER READ DATA"
4885  021006  000410                            BR      TST33           ;;GO TO NEXT TEST
4886
4887  021010  012737  051440  002224  7$:       MOV     #IBUFF+1000,E.BA ;SET EXPECTED BA
4888  021016  023737  002224  002164            CMP     E.BA,T.BA       ;TEST IF BA CORRECT
4889  021024  001401                            BEQ     5$              ;YES - SKIP
4890  021026  104052                            ERROR   52              ;BUS ADDRESS INCORRECT
4891
4892  021030                          5$:
4893
4894                                  .SBTTL  **WRITE CHECK TESTS
4895
4896                                  ;;*****************************************************************
4897                                  ;*TEST 33       WRITE CHECK OF 400 WORDS
4898                                  ;*
4899                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4900                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
4901                                  ;*      CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4902                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4903                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4904                                  ;*      A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
```

```
4905                                    ;*      AND A GOOD 32 BIT ECC.  MAKE SURE NO ERRORS SET.
4906                                    ;*
4907                                    ;;*****************************************************************
4908  021030  000004                    TST33:  SCOPE
4909  021032  012737  000012  001234            MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
4910  021040  005037  002354                    CLR     HIBITS          ;CLEAR HI ORDER BITS
4911  021044  005037  002352                    CLR     BSEDES          ;CLEAR BSE DESIRED
4912
4913  021050  005037  002344                    CLR     ECCSRC          ;CLEAR ECC SOURCE FLAG
4914  021054  005037  002344                    CLR     ECCSRC          ;CLEAR ECC SOURCE
4915  021060  004437  034026                    JSR     R4,LOADRK       ;LOAD "L" REGISTERS
4916  021064  000000                            0                       ;CYLINDER 0
4917  021066     000                            .BYTE   0               ;SECTOR 0
4918  021067     000                            .BYTE   0               ;TRACK 0
4919  021070  050440                            IBUFF                   ;BUFFER ADDRESS IBUFF
4920  021072  177400                            -400                    ;WORD COUNT -400
4921  021074  000031                            WRTCHK                  ;COMMAND WRTCHK
4922
4923  021076  004437  033566                    JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
4924
4925  021102  004437  033612                    JSR     R4,BLDDAT       ;GO BUILD DATA
4926  021106  000004                            4                       ;PATTERN 4
4927
4928  021110  012700  000400                    MOV     #400,R0         ;SET A COUNT FOR BUFFER SIZE
4929  021114  012701  050440                    MOV     #IBUFF,R1       ;SET ADDRESS OF IBUFF
4930  021120  012703  051444                    MOV     #OBUFF,R3       ;SET ADDRESS OF OBUFF
4931  021124  012321                  75$:      MOV     (R3)+,(R1)+     ;MOV PATTERN TO IBUFF
4932  021126  005300                            DEC     R0              ;DEC COUNT
4933  021130  001375                            BNE     75$             ;LOOP UNTIL PATTERN IS MOVED
4934
4935  021132  012762  100000  000000  110$:     MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
4936  021140  004437  034072                    JSR     R4,OPSTRT       ;START THE OPERATION
4937  021144  004437  025100                    JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
4938  021150  104004                            ERROR   4               ;CS1 MISCOMPARE
4939  021152  104005                            ERROR   5               ;MR1     "
4940  021154  104006                            ERROR   6               ;MR2     "
4941  021156  104007                            ERROR   7               ;MR3     "
4942
4943  021160  004437  025062                    JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
4944
4945  021164  004437  025504                    JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
4946  021170  104010                            ERROR   10              ;MR1 CONTENTS IN ERROR
4947  021172  104011                            ERROR   11              ;CS1 CONTENTS IN ERROR
4948
4949  021174  004437  026042                    JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
4950  021200  104012                            ERROR   12              ;MR1 CONTENTS IN ERROR
4951  021202  104013                            ERROR   13              ;CS1 IN ERROR AFTER SEARCH
4952  021204  104014                            ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
4953
4954  021206  004437  030276                    JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
4955  021212  104015                            ERROR   15              ;MR1 IN ERROR READING GAP
4956  021214  104016                            ERROR   16              ;CS1 IN ERROR AFTER READING GAP
4957  021216  104032                            ERROR   32              ;MR1 IN ERROR READING SYNC
4958  021220  104033                            ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
4959
4960  021222  012737  000000  002346            MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
```

```
4961  021230  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
4962  021236  004437  030774                  JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
4963  021242  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
4964  021244  104053                          ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
4965  021246  104054                          ERROR   54              ;ECC ERROR IN WRITE CHECK OP
4966  021250  104055                          ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
4967  021252  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
4968  021254  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
4969  021256  104043                          ERROR   43              ;ERR IN ECC POS COUNT
4970
4971
4972  021260  012700  002710                  MOV     #40.*37.,R0     ;SET COUNT TO EMPTY SILO
4973
4974  021264  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
4975  021272  012762  000040  000026          MOV     #DMD,RKMR1(R2)
4976  021300  005300                          DEC     R0              ;DEC COUNT
4977  021302  001370                          BNE     1$              ;LOOP UNTIL 0
4978
4979  021304  004437  034154                  JSR     R4,GETREG       ;GET 611 REGS
4980  021310  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
4981  021316  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO BIT
4982  021324  052737  000200  002220          BIS     #RDY,E.CS1      ;SET READY BIT
4983  021332  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
4984  021340  001401                          BEQ     2$              ;YES - SKIP
4985  021342  104057                          ERROR   57              ;"CS1 IN ERROR AFTER WRITE CHECK"
4986  021344                          2$:
4987
4988                                  ;;*****************************************************************
4989                                  ;*TEST 34       WRITE CHECK WITH DCK
4990                                  ;*
4991                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
4992                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
4993                                  ;*      CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
4994                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
4995                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
4996                                  ;*      A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
4997                                  ;*      AND A 32 BIT ECC INDICATING A 1 BIT BURST ERROR.
4998                                  ;*      VERIFY THAT THE POSITION REGISTER INCREMENTS PROPERLY
4999                                  ;*      AND THAT THE FINAL PATTERN AND POSITION ARE CORRECT.
5000                                  ;*      CHECK THAT DATA CHECK AND CONTROLLER ERROR ARE SET.
5001                                  ;*
5002                                  ;;*****************************************************************
5003  021344  000004                  TST34:  SCOPE
5004  021346  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
5005
5006  021354  005037  002352                  CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
5007  021360  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
5008  021364  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
5009  021370  000000                          0                       ;CYLINDER 0
5010  021372      000                         .BYTE   0               ;SECTOR 0
5011  021373      000                         .BYTE   0               ;TRACK 0
5012  021374  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
5013  021376  177400                          -400                    ;WORD COUNT -400
5014  021400  000031                          WRTCHK                  ;COMMAND WRTCHK
5015
5016  021402  004437  033566                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
```

```
5017
5018  021406  004437  033612              JSR     R4,BLDDAT       ;GO BUILD DATA
5019  021412  000005                      5                       ;PATTERN 5
5020
5021  021414  012700  000400              MOV     #400,R0         ;SET A COUNT FOR BUFFER SIZE
5022  021420  012701  050440              MOV     #IBUFF,R1       ;SET ADDRESS OF IBUFF
5023  021424  012703  051444              MOV     #OBUFF,R3       ;SET ADDRESS OF OBUFF
5024  021430  012321              75$:    MOV     (R3)+,(R1)+     ;MOV PATTERN TO IBUFF
5025  021432  005300                      DEC     R0              ;DEC COUNT
5026  021434  001375                      BNE     75$             ;LOOP UNTIL PATTERN IS MOVED
5027
5028  021436  012700  001002              MOV     #1002,R0        ;SET INDEX INTO BUFFER
5029  021442  012737  126073  052444      MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
5030  021450  012737  151052  052446      MOV     #151052,OBUFF+1002
5031  021456  012737  000001  002344      MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
5032  021464  032760  000100  051444      BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
5033  021472  001004                      BNE     100$            ;YES - SKIP
5034  021474  052760  000100  051444      BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
5035  021502  000403                      BR      101$
5036  021504  042760  000100  051444 100$: BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
5037  021512                       101$:
5038
5039  021512  012762  100000  000000 110$: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5040  021520  004437  034072              JSR     R4,OPSTRT       ;START THE OPERATION
5041  021524  004437  025100              JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
5042  021530  104004                      ERROR   4               ;CS1 MISCOMPARE
5043  021532  104005                      ERROR   5               ;MR1     "
5044  021534  104006                      ERROR   6               ;MR2     "
5045  021536  104007                      ERROR   7               ;MR3     "
5046
5047  021540  004437  025062              JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
5048
5049  021544  004437  025504              JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
5050  021550  104010                      ERROR   10              ;MR1 CONTENTS IN ERROR
5051  021552  104011                      ERROR   11              ;CS1 CONTENTS IN ERROR
5052
5053  021554  004437  026042              JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
5054  021560  104012                      ERROR   12              ;MR1 CONTENTS IN ERROR
5055  021562  104013                      ERROR   13              ;CS1 IN ERROR AFTER SEARCH
5056  021564  104014                      ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5057
5058  021566  004437  030276              JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
5059  021572  104015                      ERROR   15              ;MR1 IN ERROR READING GAP
5060  021574  104016                      ERROR   16              ;CS1 IN ERROR AFTER READING GAP
5061  021576  104032                      ERROR   32              ;MR1 IN ERROR READING SYNC
5062  021600  104033                      ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
5063
5064  021602  012737  000001  002346      MOV     #1,ECPATX       ;LOAD EXPECTED PATTERN
5065  021610  012737  004066  002350      MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
5066  021616  004437  030774              JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
5067  021622  000400                      400                     ;NUMBER OF WORDS TO SIMULATE
5068  021624  104053                      ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
5069  021626  104054                      ERROR   54              ;ECC ERROR IN WRITE CHECK OP
5070  021630  104055                      ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5071  021632  104041                      ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5072  021634  104042                      ERROR   42              ;ERR IN ECC PAT CALC.
```

```
5073  021636  104043                          ERROR   43              ;ERR IN ECC POS COUNT
5074
5075
5076  021640  012737  000020  002326          MOV     #20,BITCNT      ;SET BIT COUNT FOR CORRECTION COUNT
5077  021646  012700  000005                  MOV     #5,R0           ;SET COUNT FOR ECCPOS PASS THRU 0
5078  021652  012701  013672                  MOV     #13672,R1       ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
5079  021656  005037  001214                  CLR     $TMP5           ;CLEAR SWITCH
5080
5081  021662  004737  033414          3$:     JSR     PC,RDBIT        ;GO READ BIT
5082  021666  004737  032260                  JSR     PC,ECCGEN       ;GENERATE ECC
5083  021672  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
5084  021700  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
5085  021706  005237  002350                  INC     ECPOSX          ;BUMP POSITION EXPECTED
5086  021712  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;SET FOR COMPARE AND REPORT
5087  021720  023737  002254  002214          CMP     E.ECPT,T.ECPT   ;CHECK IF PATTERN CORRECT
5088  021726  001401                          BEQ     1$              ;YES - SKIP
5089  021730  104042                          ERROR   42
5090  021732  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS   ;CHECK IF POSITION CORRECT
5091  021740  001401                          BEQ     2$              ;YES - SKIP
5092  021742  104043                          ERROR   43
5093  021744  005237  002326          2$:     INC     BITCNT          ;BUMP BIT COUNT
5094  021750  005301                          DEC     R1              ;DEC COUNT TO ZERO IN ECCPOS
5095  021752  001343                          BNE     3$              ;NOT YET ZERO - LOOP
5096  021754  005300                          DEC     R0              ;DEC PASS THRU 0 COUNT
5097  021756  001406                          BEQ     4$              ;IF 0 - SKIP
5098  021760  012737  177777  002350          MOV     #-1,ECPOSX      ;PRESET EXPECTED POS TO GO TO ZERO
5099  021766  012701  020000                  MOV     #20000,R1       ;SET COUNT TO NEXT 0 IN ECC POS
5100  021772  000733                          BR      3$              ;GO DO LOOP AGAIN
5101
5102  021774  005737  001214          4$:     TST     $TMP5           ;TEST SWITCH
5103  022000  001012                          BNE     5$              ;ALREADY SET - SKIP
5104  022002  012701  010016                  MOV     #10016,R1       ;SET COUNT TO FINAL POSITION
5105  022006  012700  000001                  MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
5106  022012  012737  177777  002350          MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
5107  022020  005237  001214                  INC     $TMP5           ;BUMP SWITCH
5108  022024  000716                          BR      3$              ;GO DO LOOP AGAIN
5109
5110  022026  004737  033414          5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
5111  022032  004437  034154                  JSR     R4,GETREG       ;GET '611 REGISTERS
5112  022036  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
5113  022044  052737  100200  002220          BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
5114  022052  042737  000001  002220          BIC     #GO,E.CS1       ;CLEAR GO
5115  022060  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
5116  022066  001402                          BEQ     6$              ;YES - SKIP
5117  022070  104044                          ERROR   44
5118  022072  000410                          BR      7$
5119  022074  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
5120  022102  001404                          BEQ     7$              ;YES - SKIP
5121  022104  012737  100000  002234          MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
5122  022112  104045                          ERROR   45
5123
5124  022114                          7$:
5125
5126                                  ;;*****************************************************************
5127                                  ;*TEST 35       WRITE CHECK OF 18 BIT WORDS
5128                                  ;*
```

```
                                        ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                        ;*      PUT CONTROLLER IN DIAGNOSTIC MODE, ISSUE A WRITE
                                        ;*      CHECK OF 400 WORDS TO AN RK06 IN 24 SECTOR FORMAT,
                                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                        ;*      A GOOD HEADER, THE 400 WORDS TO BE WRITE CHECKED,
                                        ;*      AND A GOOD 32 BIT ECC.  MAKE SURE NO ERRORS SET.
                                        ;*
                                        ;;*****************************************************************
022114  000004                  TST35:  SCOPE
022116  012737  000001  001234        MOV     #1,$TIMES       ;;DO 1 ITERATION

022124  005037  002352                CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
022130  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
022134  005037  002344                CLR     ECCSRC          ;CLEAR ECC SOURCE
022140  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
022144  000000                        0                       ;CYLINDER 0
022146     000                        .BYTE   0               ;SECTOR 0
022147     000                        .BYTE   0               ;TRACK 0
022150  050440                        IBUFF                   ;BUFFER ADDRESS IBUFF
022152  177400                        -400                    ;WORD COUNT -400
022154  010031                        WRTCHK!CFMT                     ;COMMAND WRTCHK!CFMT

022156  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER

022162  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
022166  000004                        4                       ;PATTERN 4

022170  012700  000400                MOV     #400,R0         ;SET A COUNT FOR BUFFER SIZE
022174  012701  050440                MOV     #IBUFF,R1       ;SET ADDRESS OF IBUFF
022200  012703  051444                MOV     #OBUFF,R3       ;SET ADDRESS OF OBUFF
022204  012321                75$:    MOV     (R3)+,(R1)+     ;MOV PATTERN TO IBUFF
022206  005300                        DEC     R0              ;DEC COUNT
022210  001375                        BNE     75$             ;LOOP UNTIL PATTERN IS MOVED

022212  012762  100000  000000  110$: MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
022220  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
022224  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
022230  104004                        ERROR   4               ;CS1 MISCOMPARE
022232  104005                        ERROR   5               ;MR1     "
022234  104006                        ERROR   6               ;MR2     "
022236  104007                        ERROR   7               ;MR3     "

022240  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

022244  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
022250  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
022252  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR

022254  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
022260  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
022262  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
022264  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

022266  004437  030276                JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
022272  104015                        ERROR   15              ;MR1 IN ERROR READING GAP
```

```
5185  022274  104016                        ERROR   16              ;CS1 IN ERROR AFTER READING GAP
5186  022276  104032                        ERROR   32              ;MR1 IN ERROR READING SYNC
5187  022300  104033                        ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
5188
5189  022302  012737  000000  002346        MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
5190  022310  012737  005066  002350        MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
5191  022316  004437  030774                JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
5192  022322  000400                        400                     ;NUMBER OF WORDS TO SIMULATE
5193  022324  104053                        ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
5194  022326  104054                        ERROR   54              ;ECC ERROR IN WRITE CHECK OP
5195  022330  104055                        ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5196  022332  104041                        ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5197  022334  104042                        ERROR   42              ;ERR IN ECC PAT CALC.
5198  022336  104043                        ERROR   43              ;ERR IN ECC POS COUNT
5199
5200
5201  022340  012700  003100                MOV     #40.*40.,R0     ;SET COUNT TO EMPTY SILO
5202
5203  022344  012762  000440  000026  1$:   MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
5204  022352  012762  000040  000026        MOV     #DMD,RKMR1(R2)
5205  022360  005300                        DEC     R0              ;DEC COUNT
5206  022362  001370                        BNE     1$              ;LOOP UNTIL 0
5207
5208  022364  004437  034154                JSR     R4,GETREG       ;GET 611 REGS
5209  022370  013737  002260  002220        MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
5210  022376  042737  000001  002220        BIC     #GO,E.CS1       ;CLEAR GO BIT
5211  022404  052737  000200  002220        BIS     #RDY,E.CS1      ;SET READY BIT
5212  022412  023737  002160  002220        CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
5213  022420  001401                        BEQ     2$              ;YES - SKIP
5214  022422  104057                        ERROR   57              ;"CS1 IN ERROR AFTER WRITE CHECK"
5215  022424                          2$:
5216                                  ;;*****************************************************************
5217                                  ;*TEST 36       WRITE CHECK OF A PARTIAL SECTOR
5218                                  ;*
5219                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5220                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
5221                                  ;*      CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
5222                                  ;*      CYLINER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
5223                                  ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
5224                                  ;*      A GOOD HEADER, TAKE ONE WORD TO BE WRITE-CHECKED, 377 WORDS
5225                                  ;*      NOT TO BE WRITE CHECKED, AND A GOOD 32 BIT ECC.
5226                                  ;*      MAKE SURE NO ERRORS SET.
5227                                  ;*
5228                                  ;;*****************************************************************
5229  022424  000004                  TST36:  SCOPE
5230  022426  012737  000001  001234        MOV     #1,$TIMES       ;;DO 1 ITERATION
5231
5232  022434  005037  002352                CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
5233  022440  005037  002354                CLR     HIBITS          ;CLEAR HI ORDER BITS
5234  022444  005037  002344                CLR     ECCSRC          ;CLEAR ECC SOURCE
5235  022450  004437  034026                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
5236  022454  000000                        0                       ;CYLINDER 0
5237  022456     000                        .BYTE   0               ;SECTOR 0
5238  022457     000                        .BYTE   0               ;TRACK 0
5239  022460  050440                        IBUFF                   ;BUFFER ADDRESS IBUFF
5240  022462  177777                        -1                      ;WORD COUNT -1
```

```
5241  022464  000031                        WRTCHK                  ;COMMAND WRTCHK
5242
5243  022466  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
5244
5245  022472  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
5246  022476  000006                        6                       ;PATTERN 6
5247
5248  022500  012700  000400                MOV     #400,R0         ;SET A COUNT FOR BUFFER SIZE
5249  022504  012701  050440                MOV     #IBUFF,R1       ;SET ADDRESS OF IBUFF
5250  022510  012703  051444                MOV     #OBUFF,R3       ;SET ADDRESS OF OBUFF
5251  022514  012321                75$:    MOV     (R3)+,(R1)+     ;MOV PATTERN TO IBUFF
5252  022516  005300                        DEC     R0              ;DEC COUNT
5253  022520  001375                        BNE     75$             ;LOOP UNTIL PATTERN IS MOVED
5254
5255  022522  012762  100000  000000  110$:  MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5256  022530  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
5257  022534  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
5258  022540  104004                        ERROR   4               ;CS1 MISCOMPARE
5259  022542  104005                        ERROR   5               ;MR1     "
5260  022544  104006                        ERROR   6               ;MR2     "
5261  022546  104007                        ERROR   7               ;MR3     "
5262
5263  022550  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
5264
5265  022554  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
5266  022560  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
5267  022562  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
5268
5269  022564  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
5270  022570  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
5271  022572  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
5272  022574  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5273
5274  022576  004437  030276                JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
5275  022602  104015                        ERROR   15              ;MR1 IN ERROR READING GAP
5276  022604  104016                        ERROR   16              ;CS1 IN ERROR AFTER READING GAP
5277  022606  104032                        ERROR   32              ;MR1 IN ERROR READING SYNC
5278  022610  104033                        ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
5279
5280  022612  012737  000000  002346        MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
5281  022620  012737  004066  002350        MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
5282  022626  004437  030774                JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
5283  022632  000400                        400                     ;NUMBER OF WORDS TO SIMULATE
5284  022634  104053                        ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
5285  022636  104054                        ERROR   54              ;ECC ERROR IN WRITE CHECK OP
5286  022640  104055                        ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5287  022642  104041                        ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5288  022644  104042                        ERROR   42              ;ERR IN ECC PAT CALC.
5289  022646  104043                        ERROR   43              ;ERR IN ECC POS COUNT
5290
5291
5292  022650  013700  002164                MOV     T.BA,R0         ;GET ACTUAL BA
5293  022654  162700  050440                SUB     #IBUFF,R0       ;SUB START VALUE OF BA
5294  022660  022700  000002                CMP     #2,R0           ;TEST IF CORRECT NUM OF WORDS XFER
5295  022664  001404                        BEQ     4$              ;YES - SKIP
5296  022666  012737  050442  002224        MOV     #IBUFF+2,E.BA   ;SET EXPECTED BA
```

```
                                        ERROR   46              ;REPORT XFER LENGTH ERROR

                        4$:     MOV     #8.*4.,R1       ;SET COUNT TO END OF OPERATION
                        3$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
                                MOV     #DMD,RKMR1(R2)
                                DEC     R1              ;DEC COUNT
                                BNE     3$              ;LOOP UNTIL ZERO

                        1$:     JSR     R4,GETREG       ;GET 611 REGS
                                MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
                                BIC     #GO,E.CS1       ;CLEAR GO BIT
                                BIS     #RDY,E.CS1      ;SET READY BIT
                                CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
                                BEQ     2$              ;YES - SKIP
                                ERROR   57              ;"CS1 IN ERROR AFTER WRITE CHECK"
                        2$:

;;*****************************************************************
;*TEST 37        WRITE CHECK ERROR (PART 1)
;*
;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
;*      CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
;*      CLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
;*      A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 000000.
;*      WRITE CHECKED DATA IS 000001.  MAKE SURE WRITE CHECK ERROR SETS.
;*      REPEAT USING EACH OF THE FOLLOWING CONFIGURATIONS
;*      AS WRITE CHECKED DATA:
;*
;*      000002  000040  001000  020000
;*      000004  000100  002000  040000
;*      000010  000200  004000  100000
;*      000020  000400  010000
;*
;;*****************************************************************
TST37:  SCOPE
                                MOV     #1,$TIMES       ;;DO 1 ITERATION

                                CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
                                CLR     HIBITS          ;CLEAR HI ORDER BITS
                                CLR     ECCSRC          ;CLEAR ECC SOURCE
                                JSR     R4,LOADRK       ;LOAD "L" REGISTERS
                                0                       ;CYLINDER 0
                                .BYTE   0               ;SECTOR 0
                                .BYTE   0               ;TRACK 0
                                IBUFF                   ;BUFFER ADDRESS IBUFF
                                -1                      ;WORD COUNT -1
                                WRTCHK                  ;COMMAND WRTCHK

                                MOV     #16.,R5         ;SET PATTERN COUNT
                                CLR     OBUFF           ;CLEAR FIRST WORD OF OBUFF
                                MOV     #1,IBUFF        ;SET IN FIRST PATTERN
                                BIT     #BIT0,IBUFF     ;IS BIT 0 SET
                                BNE     112$            ;YES - PATTERN SHOULD BE 0
```

```
5353  023052  012737  177777  051444          MOV     #-1,OBUFF       ;ELSE PATTERN SHOULD BE ALL ONES
5354  023060  012737  023066  001110  112$:   MOV     #110$,$LPERR    ;SET LOCAL LOOP ON ERROR
5355
5356  023066  012762  100000  000000  110$:   MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5357  023074  004437  034072                  JSR     R4,OPSTRT       ;START THE OPERATION
5358  023100  004437  025100                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
5359  023104  104004                          ERROR   4               ;CS1 MISCOMPARE
5360  023106  104005                          ERROR   5               ;MR1     "
5361  023110  104006                          ERROR   6               ;MR2     "
5362  023112  104007                          ERROR   7               ;MR3     "
5363
5364  023114  004437  025062                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
5365
5366  023120  004437  025504                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
5367  023124  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
5368  023126  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR
5369
5370  023130  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
5371  023134  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
5372  023136  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
5373  023140  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5374
5375  023142  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
5376  023146  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
5377  023150  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
5378  023152  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
5379  023154  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
5380
5381  023156  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
5382  023164  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
5383  023172  004437  030774                  JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
5384  023176  000001                          1                       ;NUMBER OF WORDS TO SIMULATE
5385  023200  104053                          ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
5386  023202  104054                          ERROR   54              ;ECC ERROR IN WRITE CHECK OP
5387  023204  104055                          ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5388  023206  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5389  023210  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
5390  023212  104043                          ERROR   43              ;ERR IN ECC POS COUNT
5391
5392  023214  012701  000040                  MOV     #8.*4.,R1       ;SET COUNT
5393  023220  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
5394  023226  012762  000040  000026          MOV     #DMD,RKMR1(R2)
5395  023234  005301                          DEC     R1              ;DEC COUNT
5396  023236  001370                          BNE     1$              ;LOOP UNTIL ZERO
5397
5398  023240  004437  034154                  JSR     R4,GETREG       ;GET REGS
5399  023244  032737  040000  002170          BIT     #WCE,T.CS2      ;TEST IF WCE SET
5400  023252  001012                          BNE     2$              ;YES - SKIP
5401  023254  013737  002170  001162          MOV     T.CS2,$REG0     ;SET CS2 FOR REPORT
5402  023262  013737  050440  001164          MOV     IBUFF,$REG1     ;SET WORD READ
5403  023270  013737  051444  001166          MOV     OBUFF,$REG2     ;SET WORD FROM BUFFER
5404  023276  104060                          ERROR   60              ;"WCE FAILED TO SET"
5405
5406  023300  104415                  2$:     SCOP1                   ;LOCAL LOOP ON ERROR
5407
5408  023302  006337  050440                  ASL     IBUFF           ;SHIFT FOR NEXT TEST PATTERN
```

```
                                        DEC     R5              ;DEC PATTERN COUNT
                                        BNE     110$            ;LOOP UNTIL ALL PATTERNS TESTED

                        ;;*****************************************************************
                        ;*TEST 40       WRITE CHECK ERROR (PART 2)
                        ;*
                        ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                        ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                        ;*      CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
                        ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                        ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE
                        ;*      A GOOD HEADER, AND ONE WORD OF DATA CONSISTING OF 177777.
                        ;*      WRITE CHECKED DATA IS 177776.  MAKE SURE WRITE
                        ;*      CHECK ERROR SETS.  REPEAT USING EACH OF THE FOLLOWING
                        ;*      OCNFIGURATION AS WRITE CHECKED DATA:
                        ;*
                        ;*      177775  177737  176777  157777
                        ;*      177773  177677  175777  137777
                        ;*      177767  177577  173777  077777
                        ;*      177757  177377  167777
                        ;*
                        ;;*****************************************************************
023312  000004          TST40:  SCOPE
023314  012737  000001  001234  MOV     #1,$TIMES       ;;DO 1 ITERATION

023322  005037  002352          CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
023326  005037  002354          CLR     HIBITS          ;CLEAR HI ORDER BITS
023332  005037  002344          CLR     ECCSRC          ;CLEAR ECC SOURCE
023336  004437  034026          JSR     R4,LOADRK       ;LOAD "L" REGISTERS
023342  000000                  0                       ;CYLINDER 0
023344      000                 .BYTE   0               ;SECTOR 0
023345      000                 .BYTE   0               ;TRACK 0
023346  050440                  IBUFF                   ;BUFFER ADDRESS IBUFF
023350  177777                  -1                      ;WORD COUNT -1
023352  000031                  WRTCHK                  ;COMMAND WRTCHK

023354  012705  000020          MOV     #16.,R5         ;SET PATTERN COUNT
023360  005037  051444          CLR     OBUFF           ;CLEAR FIRST WORD OF OBUFF
023364  012737  177776  050440  MOV     #177776,IBUFF   ;SET IN FIRST PATTERN
023372  032737  000001  050440  BIT     #BIT0,IBUFF     ;IS BIT 0 SET
023400  001003                  BNE     112$            ;YES - PATTERN SHOULD BE 0
023402  012737  177777  051444  MOV     #-1,OBUFF       ;ELSE PATTERN SHOULD BE ALL ONES
023410  012737  023416  001110  112$:  MOV     #110$,$LPERR    ;SET LOCAL LOOP ON ERROR

023416  012762  100000  000000  110$:  MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
023424  004437  034072          JSR     R4,OPSTRT       ;START THE OPERATION
023430  004437  025100          JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
023434  104004                  ERROR   4               ;CS1 MISCOMPARE
023436  104005                  ERROR   5               ;MR1     "
023440  104006                  ERROR   6               ;MR2     "
023442  104007                  ERROR   7               ;MR3     "

023444  004437  025062          JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

023450  004437  025504          JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
023454  104010                  ERROR   10              ;MR1 CONTENTS IN ERROR
```

```
005465  023456  104011                          ERROR   11              ;CSI CONTENTS IN ERROR
005466
005467  023460  004437  026042                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
005468  023464  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
005469  023466  104013                          ERROR   13              ;CSI IN ERROR AFTER SEARCH
005470  023470  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
005471
005472  023472  004437  030276                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
005473  023476  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
005474  023500  104016                          ERROR   16              ;CSI IN ERROR AFTER READING GAP
005475  023502  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
005476  023504  104033                          ERROR   33              ;CSI IN ERROR AFTER READING SYNC
005477
005478  023506  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
005479  023514  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
005480  023522  004437  030774                  JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
005481  023526  000001                          1                       ;NUMBER OF WORDS TO SIMULATE
005482  023530  104053                          ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
005483  023532  104054                          ERROR   54              ;ECC ERROR IN WRITE CHECK OP
005484  023534  104055                          ERROR   55              ;CSI ERROR AFTER WRT CHK OR ECC READ
005485  023536  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
005486  023540  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
005487  023542  104043                          ERROR   43              ;ERR IN ECC POS COUNT
005488
005489  023544  012701  000040                  MOV     #8.*4.,R1       ;SET COUNT
005490  023550  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
005491  023556  012762  000040  000026          MOV     #DMD,RKMR1(R2)
005492  023564  005301                          DEC     R1              ;DEC COUNT
005493  023566  001370                          BNE     1$              ;LOOP UNTIL ZERO
005494
005495  023570  004437  034154                  JSR     R4,GETREG       ;GET REGS
005496  023574  032737  040000  002170          BIT     #WCE,T.CS2      ;TEST IF WCE SET
005497  023602  001012                          BNE     2$              ;YES - SKIP
005498  023604  013737  002170  001162          MOV     T.CS2,$REG0     ;SET CS2 FOR REPORT
005499  023612  013737  050440  001164          MOV     IBUFF,$REG1     ;SET WORD READ
005500  023620  013737  051444  001166          MOV     OBUFF,$REG2     ;SET WORD FROM BUFFER
005501  023626  104060                          ERROR   60              ;"WCE FAILED TO SET"
005502
005503  023630  104415                  2$:     SCOP1                   ;LOCAL LOOP ON ERROR
005504
005505  023632  006337  050440                  ASL     IBUFF           ;SHIFT FOR NEXT TEST PATTERN
005506  023636  005537  050440                  ADC     IBUFF           ;PUT CARRY BACK IN
005507  023642  005305                          DEC     R5              ;DEC PATTERN COUNT
005508  023644  001264                          BNE     110$            ;LOOP UNTIL ALL PATTERNS TESTED
005509
005510                          ;;*****************************************************************
005511                          ;*TEST 41        WRITE CHECK ERROR (PART 3)
005512                          ;*
005513                          ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
005514                          ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
005515                          ;*      CHECK OF 1 WORD TO AN RK06 IN 24 SECTOR FORMAT,
005516                          ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
005517                          ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
005518                          ;*      A GOOD HEADER, AND ONE WORD OF DATA CONSISTING
005519                          ;*      OF 200000.  WRITE CHECK DATA IS 000000.  MAKE SURE
005520                          ;*      WRITE CHECK ERROR SETS.  REPEAT USING 400000 AS
```

```
                                        ;*      SIMULATED DATA.
                                        ;*
                                        ;;*********************************************************************
023646  000004                          TST41:  SCOPE
023650  012737  000001  001234                  MOV     #1,$TIMES       ;;DO 1 ITERATION
023656  005037  002352                          CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
023662  005037  002344                          CLR     ECCSRC          ;CLEAR ECC SOURCE
023666  004437  034026                          JSR     R4,LOADRK       ;LOAD "L" REGISTERS
023672  000000                                  0                       ;CYLINDER 0
023674     000                                  .BYTE   0               ;SECTOR 0
023675     000                                  .BYTE   0               ;TRACK 0
023676  050440                                  IBUFF                   ;BUFFER ADDRESS IBUFF
023700  177777                                  -1                      ;WORD COUNT -1
023702  010031                                  WRTCHK!CFMT                     ;COMMAND WRTCHK!CFMT
023704  012737  040000  002354                  MOV     #40000,HIBITS   ;SET HIBITS FOR PATTERN OF 200000
023712  005037  050440                          CLR     IBUFF           ;SET LOW BITS FOR READ
023716  005037  051444                          CLR     OBUFF           ;SET 0 IN BUFFER
023722  012737  023730  001110  112$:           MOV     #110$,$LPERR    ;SET LOCAL LOOP ON ERROR
023730  012762  100000  000000  110$:           MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
023736  012700  002000                          MOV     #2000,R0        ;SET STALL COUNT
023742  005300                          4$:     DEC     R0              ;DEC COUNT
023744  001376                                  BNE     4$              ;LOOP UNTIL ZERO
023746  004437  034072                          JSR     R4,OPSTRT       ;START THE OPERATION
023752  004437  025100                          JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
023756  104004                                  ERROR   4               ;CS1 MISCOMPARE
023760  104005                                  ERROR   5               ;MR1    "
023762  104006                                  ERROR   6               ;MR2    "
023764  104007                                  ERROR   7               ;MR3    "
023766  004437  025062                          JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
023772  004437  025504                          JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
023776  104010                                  ERROR   10              ;MR1 CONTENTS IN ERROR
024000  104011                                  ERROR   11              ;CS1 CONTENTS IN ERROR
024002  004437  026042                          JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
024006  104012                                  ERROR   12              ;MR1 CONTENTS IN ERROR
024010  104013                                  ERROR   13              ;CS1 IN ERROR AFTER SEARCH
024012  104014                                  ERROR   14              ;RKDCYL OR RKDA IN ERROR OFTER SEARCH
024014  004437  030276                          JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
024020  104015                                  ERROR   15              ;MR1 IN ERROR READING GAP
024022  104016                                  ERROR   16              ;CS1 IN ERROR AFTER READING GAP
024024  104032                                  ERROR   32              ;MR1 IN ERROR READING SYNC
024026  104033                                  ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
024030  012737  000000  002346                  MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
024036  012737  005066  002350                  MOV     #5066,ECPOSX    ;LOAD EXPECTED POSITION
024044  004437  030774                          JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
024050  000001                                  1                       ;NUMBER OF WORDS TO SIMULATE
024052  104053                                  ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
024054  104054                                  ERROR   54              ;ECC ERROR IN WRITE CHECK OP
```

```
5577  024056  104055                          ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5578  024060  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5579  024062  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
5580  024064  104043                          ERROR   43              ;ERR IN ECC POS COUNT
5581
5582  024066  012701  000040                  MOV     #8.*4.,R1       ;SET COUNT
5583  024072  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
5584  024100  012762  000040  000026          MOV     #DMD,RKMR1(R2)
5585  024106  005301                          DEC     R1              ;DEC COUNT
5586  024110  001370                          BNE     1$              ;LOOP UNTIL ZERO
5587
5588  024112  004437  034154                  JSR     R4,GETREG       ;GET REGS
5589  024116  032737  040000  002170          BIT     #WCE,T.CS2      ;TEST IF WCE SET
5590  024124  001025                          BNE     2$              ;YES - SKIP
5591  024126  013737  002170  001162          MOV     T.CS2,$REG0     ;SET CS2 FOR REPORT
5592  024134  013737  050440  001164          MOV     IBUFF,$REG1     ;SET WORD READ
5593  024142  013737  051444  001166          MOV     OBUFF,$REG2     ;SET WORD FROM BUFFER
5594  024150  005037  001170                  CLR     $REG3           ;SET UP FOR REPORT
5595  024154  013737  002354  001172          MOV     HIBITS,$REG4
5596  024162  012700  000016                  MOV     #14.,R0
5597  024166  006037  001172          3$:     ROR     $REG4
5598  024172  005300                          DEC     R0
5599  024174  001374                          BNE     3$
5600  024176  104063                          ERROR   63              ;"WCE FAILED TO SET"
5601
5602  024200  104415                  2$:     SCOP1                   ;LOCAL LOOP ON ERROR
5603
5604  024202  023727  002354  100000          CMP     HIBITS,#100000  ;TEST IF SECOND PATTERN DONE
5605  024210  001404                          BEQ     5$              ;YES - EXIT
5606  024212  012737  100000  002354          MOV     #100000,HIBITS  ;SET FOR PATTERN 400000
5607  024220  000643                          BR      110$            ;LOOP UNTIL ALL PATTERNS TESTED
5608  024222                          5$:
5609
5610                                  ;;*****************************************************************
5611                                  ;*TEST 42       WRITE CHECK ERROR (PART 4)
5612                                  ;*
5613                                  ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
5614                                  ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
5615                                  ;*      CHECK OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
5616                                  ;*      CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
5617                                  ;*      AND DRIVE CLEAR MESSAGES SIMULATE A SECTOR PULSE,
5618                                  ;*      A GOOD HEADER, 125 GOOD WORDS, AND A BAD WORD.
5619                                  ;*      MAKE SURE WRITE CHECK ERROR SET.  CHECK WORD COUNT
5620                                  ;*      AND BUS ADDRESS.
5621                                  ;*
5622                                  ;;*****************************************************************
5623  024222  000004                  TST42:  SCOPE
5624  024224  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
5625
5626  024232  005037  002354                  CLR     HIBITS          ;CLEAR HI BITS
5627  024236  005037  002352                  CLR     BSEDES          ;CLEAR BAD SECTOR DESIRED
5628  024242  004437  034026                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
5629  024246  000000                          0                       ;CYLINDER 0
5630  024250     000                          .BYTE   0               ;SECTOR 0
5631  024251     000                          .BYTE   0               ;TRACK 0
5632  024252  050440                          IBUFF                   ;BUFFER ADDRESS IBUFF
```

```
5633  024254  177400                        -400                    ;WORD COUNT -400
5634  024256  000031                        WRTCHK                  ;COMMAND WRTCHK
5635
5636  024260  004437  033566                JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
5637
5638  024264  004437  033612                JSR     R4,BLDDAT       ;GO BUILD DATA
5639  024270  000005                        5                       ;PATTERN 5
5640
5641  024272  012700  000400                MOV     #400,R0         ;SET A COUNT FOR BUFFER SIZE
5642  024276  012701  050440                MOV     #IBUFF,R1       ;SET ADDRESS OF IBUFF
5643  024302  012703  051444                MOV     #OBUFF,R3       ;SET ADDRESS OF OBUFF
5644  024306  012321                75$:    MOV     (R3)+,(R1)+     ;MOV PATTERN TO IBUFF
5645  024310  005300                        DEC     R0              ;DEC COUNT
5646  024312  001375                        BNE     75$             ;LOOP UNTIL PATTERN IS MOVED
5647  024314  012700  000530                MOV     #530,R0         ;SET INDEX INTO BUFFER
5648
5649  024320  012737  126073  052444        MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
5650  024326  012737  151052  052446        MOV     #151052,OBUFF+1002
5651  024334  012737  000001  002344        MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
5652  024342  032760  000100  051444        BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
5653  024350  001004                        BNE     100$            ;YES - SKIP
5654  024352  052760  000100  051444        BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
5655  024360  000403                        BR      101$
5656  024362  042760  000100  051444  100$:  BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
5657  024370                        101$:
5658
5659  024370  012762  100000  000000  110$:  MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
5660  024376  004437  034072                JSR     R4,OPSTRT       ;START THE OPERATION
5661  024402  004437  025100                JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
5662  024406  104004                        ERROR   4               ;CS1 MISCOMPARE
5663  024410  104005                        ERROR   5               ;MR1     "
5664  024412  104006                        ERROR   6               ;MR2     "
5665  024414  104007                        ERROR   7               ;MR3     "
5666
5667  024416  004437  025062                JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
5668
5669  024422  004437  025504                JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
5670  024426  104010                        ERROR   10              ;MR1 CONTENTS IN ERROR
5671  024430  104011                        ERROR   11              ;CS1 CONTENTS IN ERROR
5672
5673  024432  004437  026042                JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
5674  024436  104012                        ERROR   12              ;MR1 CONTENTS IN ERROR
5675  024440  104013                        ERROR   13              ;CS1 IN ERROR AFTER SEARCH
5676  024442  104014                        ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
5677
5678  024444  004437  030276                JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
5679  024450  104015                        ERROR   15              ;MR1 IN ERROR READING GAP
5680  024452  104016                        ERROR   16              ;CS1 IN ERROR AFTER READING GAP
5681  024454  104032                        ERROR   32              ;MR1 IN ERROR READING SYNC
5682  024456  104033                        ERROR   33              ;CS1 IN ERROR AFTER READING SYNC
5683
5684  024460  012737  000000  002346        MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
5685  024466  012737  004066  002350        MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
5686  024474  004437  030774                JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
5687  024500  000310                        310                     ;NUMBER OF WORDS TO SIMULATE
5688  024502  104053                        ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
```

```
5689  024504  104054                          ERROR   54              ;ECC ERROR IN WRITE CHECK OP
5690  024506  104055                          ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
5691  024510  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
5692  024512  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
5693  024514  104043                          ERROR   43              ;ERR IN ECC POS COUNT
5694
5695
5696  024516  012701  000020                  MOV     #4.*4.,R1       ;SET COUNT TO END OF OPERATION
5697  024522  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;RUN CLOCK
5698  024530  012762  000040  000026          MOV     #DMD,RKMR1(R2)
5699  024536  005301                          DEC     R1              ;DEC COUNT
5700  024540  001370                          BNE     1$              ;LOOP UNTIL ZERO
5701
5702  024542  004437  034154                  JSR     R4,GETREG       ;GET '611 REGS
5703  024546  032737  040000  002170          BIT     #WCE,T.CS2      ;TEST IF WRITE CHECK SET
5704  024554  001013                          BNE     2$              ;YES - SKIP
5705  024556  013737  002170  001162          MOV     T.CS2,$REG0     ;SET UP FOR REPORT
5706  024564  016037  050440  001164          MOV     IBUFF(R0),$REG1 ;BUFFER WORD
5707  024572  016037  051444  001166          MOV     OBUFF(R0),$REG2 ;SIMULATED WORD
5708  024600  104060                          ERROR   60              ;WCE NOT SET ERROR
5709  024602  000423                          BR      4$              ;EXIT
5710
5711  024604  012737  050442  002224  2$:     MOV     #IBUFF+2,E.BA   ;SET UP EXPECTED BA
5712  024612  060037  002224                  ADD     R0,E.BA         ;ADD IN THE NUMBER OF WORDS TO THE ERROR
5713  024616  023737  002224  002164          CMP     E.BA,T.BA       ;CHECK IF CORRECT
5714  024624  001402                          BEQ     3$              ;YES - SKIP
5715  024626  104061                          ERROR   61              ;BUS ADDRESS ERROR
5716  024630  000410                          BR      4$              ;EXIT
5717
5718  024632  012737  177655  002222  3$:     MOV     #-123,E.WC      ;SET EXPECTED WORD COUNT
5719  024640  023737  002162  002222          CMP     T.WC,E.WC       ;CHECK IF CORRECT
5720  024646  001401                          BEQ     4$              ;YES - SKIP
5721  024650  104062                          ERROR   62              ;WORD COUNT ERROR
5722
5723  024652                          4$:
5724
```

```
                                .SBTTL  END OF PASS ROUTINE

                                ;;**********************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO NEWPAS

024652                          $EOP:
024652  000004                          SCOPE
024654  005037  001102                 CLR     $TSTNM          ;;ZERO THE TEST NUMBER
024660  005037  001234                 CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
024664  005237  001256                 INC     $PASS           ;;INCREMENT THE PASS NUMBER
024670  042737  100000  001256         BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
024676  005327                         DEC     (PC)+           ;;LOOP?
024700  000001                  $EOPCT: .WORD   1
024702  003063                         BGT     $DOAGN          ;;YES
024704  012737                         MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
024706  000001                  $ENDCT: .WORD   1
024710  024700                          $EOPCT
024712  104401  024720                 TYPE    ,65$            ;;TYPE ASCIZ STRING
024716  000407                         BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
024736                          64$:
024736  013746  001256                 MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE PASS NUMBER
024742  104405                         TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
024744  104401  024752                 TYPE    ,67$            ;;TYPE ASCIZ STRING
024750  000421                         BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
025014                          66$:
025014  013746  001112                 MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                ;;TOTAL NUMBER OF ERRORS
025020  104405                         TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
025022  104401  001245                 TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
025026  005037  001112                 CLR     $ERTTL          ;;CLEAR ERROR TOTAL
025032  013700  000042         $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
025036  001405                         BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
025040  000005                         RESET                   ;;CLEAR THE WORLD
025042  004710                 $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
025044  000240                         NOP                     ;;SAVE ROOM
025046  000240                         NOP                     ;;FOR
025050  000240                         NOP                     ;;ACT11
025052                          $DOAGN:
025052  000137                         JMP     @(PC)+          ;;RETURN
025054  003376                 $RTNAD: .WORD   NEWPAS
025056     377     377     000 $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
        025062                          .EVEN

                                .SBTTL  DIAGNOSTIC MODE SECTOR PULSE
                                ;*      THE SECTOR PULSE IS PROVIDED BY THIS ROUTINE
025062  012762  000140  000026  DSECTR: MOV     #DMD!MSP,RKMR1(R2)      ;SET SECTOR PULE
025070  012762  000040  000026         MOV     #DMD,RKMR1(R2)  ;RESET SECTOR PULSE
025076  000204                         RTS     R4
```

```
                                .SBTTL  DIAGNOSTIC MODE IMPLIED SEEK
                                ;*      THE CONTROLLER IS CLOCKED THROUGH THE SEEK MESSAGES,
                                ;*      GETTING STATUS, AND CLEARING THE DRIVE. SINCE THIS REQUIRES A
                                ;*      DIFFERENT COUNT FOR WRITE DATA AND READ DATA/WRITE CHECK,
                                ;*      THE ROUTINE LOOKS AT THE COMMAND THAT WAS
                                ;*      STARTED (L.CS1) AND SETS THE CLOCK COUNT ACCORDINGLY.
                                ;*
                                ;*      AT THE END OF THE IMPLIED SEEK THE CONTENTS OF CS1, MR1, MR2
                                ;*      AND MR3 ARE CHECKED TO VERIFY THE OPERATION. ALL DISCREPENCIES
                                ;*      ARE REPORTED IN THE ORDER LISTED.
                                ;*
                                ;*      CALL:
                                ;*      JSR     R4;DISEEK
                                ;*      ERROR   4       ;CS1 MISCOMPARE ERROR
                                ;*      ERROR   5       ;MR1 MISCOMPARE ERROR
                                ;*      ERROR   6       ;MR2 MISCOMPARE ERROR
                                ;*      ERROR   7       ;MR3 MISCOMPARE ERRON

025100                          DISEEK:
025100  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025102  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025104  032737  000002  002260          BIT     #BIT1,L.CS1     ;CHECK IF WRITE DATA
025112  001003                          BNE     10$             ;YES - SKIP
025114  012700  000316                  MOV     #51.*4+2,R0     ;SET COUNT FOR READ OR WRITE CHECK
025120  000412                          BR      1$

025122  012700  004612          10$:    MOV     #66.*37.,R0     ;COUNT FOR CLOCK THROUGH SILO LOAD
025126  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF 22 SECTOR MODE
025134  001402                          BEQ     11$             ;NO - SKIP
025136  012700  005222                  MOV     #66.*41.,R0     ;ELSE CHANGE FOR 18 BIT WORDS
025142  062700  000016          11$:    ADD     #4*3+2,R0       ;AND SHIFT REGISTER LOAD

025146  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;SET CLOCK
025154  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;CLEAR CLOCK
025162  005300                          DEC     R0              ;LOOP UNTIL R0 IS ZERO
025164  001370                          BNE     1$

025166  013703  002274                  MOV     L.DCYL,R3       ;GET DESIRED CYLINDER
025172  004437  034242                  JSR     R4,FSBLVV       ;REVERSE ORDER OF BITS
025176  010337  002246                  MOV     R3,E.MR2        ;STORE AS EXPECTED MR2
025202  113703  002267                  MOVB    L.DT,R3         ;GET DESIRED TRACK/SECTOR
025206  042703  177400                  BIC     #177400,R3      ;CLEAR UNUSED BITS
025212  012700  000005                  MOV     #5,R0           ;SET SHIFT COUNT
025216  006303                  15$:    ASL     R3              ;SHIFT FOR TRACK ALIGNMENT
025220  005300                          DEC     R0              ;DEC COUNT
025222  001375                          BNE     15$             ;LOOP UNTIL ZERO
025224  153703  002266                  BISB    L.DS,R3         ;INSERT SECTOR NUMBER
025230  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
025236  001402                          BEQ     16$             ;NO - SKIP
025240  052703  001000                  BIS     #BIT9,R3        ;ELSE SET FORMAT BIT IN HDR WRD
025244  004437  034242          16$:    JSR     R4,FSBLVV       ;REVERSE BIT ORDER
025250  010337  002250                  MOV     R3,E.MR3        ;STORE AS EXPECTED MR3

025254  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
025262  012737  022040  002244          MOV     #DMD!ECCW!MEWD,E.MR1 ;SET UP EXPECTED MR1
025270  004437  034154                  JSR     R4,GETREG       ;GET RK REGS
```

```
5837
5838  025274  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK CS1
5839  025302  001411                          BEQ     2$              ; OK - SKIP
5840  025304  012737  025326  001236          MOV     #2$,$ESCAPE     ;SET ESCAPE TO RETURN FOR
5841                                                                  ;REMAINING TESTS
5842  025312  032777  001000  153620          BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
5843  025320  001056                          BNE     9$              ;YES - SKIP
5844
5845  025322  010446                  4$:     MOV     R4,-(SP)        ;SET STACK FOR RETURN TO SUB-ROUTINE
5846  025324  000204                          RTS     R4
5847
5848  025326  005724                  2$:     TST     (R4)+           ;BUMP TO NEXT ERROR CALL
5849  025330  023737  002244  002204          CMP     E.MR1,T.MR1     ;CHECK MR1
5850  025336  001410                          BEQ     3$              ;OK - SKIP
5851  025340  032777  001000  153572          BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
5852  025346  001043                          BNE     9$              ;YES - SKIP
5853  025350  012737  025360  001236          MOV     #3$,$ESCAPE     ;SET ESCAPE FOR RETURN TO SUB-ROUTINE
5854  025356  000761                          BR      4$
5855
5856  025360  005724                  3$:     TST     (R4)+           ;SAME OPERATION AS ABOVE TO
5857  025362  023737  002246  002206          CMP     E.MR2,T.MR2     ;CHECK MR2
5858  025370  001410                          BEQ     5$
5859  025372  032777  001000  153540          BIT     #BIT9,@SWR
5860  025400  001026                          BNE     9$
5861  025402  012737  025412  001236          MOV     #5$,$ESCAPE
5862  025410  000744                          BR      4$
5863
5864  025412  005724                  5$:     TST     (R4)+           ;SAME OPERATION AS ABOVE TO
5865  025414  023737  002250  002210          CMP     E.MR3,T.MR3     ;CHECK MR3
5866  025422  001414                          BEQ     8$
5867  025424  032777  001000  153506          BIT     #BIT9,@SWR
5868  025432  001011                          BNE     9$
5869  025434  012737  025444  001236          MOV     #6$,$ESCAPE
5870  025442  000727                          BR      4$
5871
5872  025444  032777  001000  153466  6$:     BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
5873  025452  001001                          BNE     9$              ;YES - SKIP
5874  025454  005724                  8$:     TST     (R4)+           ;BUMP TO NO ERROR RETURN
5875  025456  005037  001236          9$:     CLR     $ESCAPE         ;CLEAR ESCAPE
5876  025462  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
5877  025464  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
5878  025466  000204                          RTS     R4
5879
5880  025470  005037  001236          7$:     CLR     $ESCAPE         ;CLEAR ESCAPE
5881  025474  012706  001100                  MOV     #STACK,SP       ;CLEAN OFF STACK
5882  025500  000177  153404                  JMP     @$LPERR         ;GO TO LOOP START
5883
5884                                  .SBTTL  DIAGNOSTIC MODE READ SYNC PATTERN (PREAMBLE)
5885                                  ;*      THE 255 "0" BITS AND 1 "1" BIT ARE CLOCKED THROUTH THE READ.
5886                                  ;*      THE CONTENTS OF MR1 ARE MONITORED TO WATCH FOR AN ERROR.
5887                                  ;*      AT THE END OF THE SYNC READ, CS1 IS CHECKED TO INSURE IT
5888                                  ;*      DID NOT GET CHANGED.
5889                                  ;*
5890                                  ;*      CALL:
5891                                  ;*      JSR     R4,DRSYNC
5892                                  ;*      ERROR   10      ;MR1 CONTENTS IN ERROR
```

```
                                        ;*      ERROR   11      ;CS1 CONTENTS IN ERROR
                                        ;*
                                        ;*      IF AN ERROR IS DETECTED, THE SUBROUTINE WILL SET UP TO ABORT
                                        ;*      THE PRESENT TEST UNLESS LOOP ON ERROR IS SET.

025504                                  DRSYNC:
025504  010046                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025506  005037  002326                        CLR     BITCNT          ;CLEAR BIT COUNT
025512  005037  001236                        CLR     $ESCAPE         ;CLEAR ESCAPE
025516  012700  000200                        MOV     #128.,R0        ;SET COUNT TO CLOCK TO READ GATE
025522  005037  002320                        CLR     PR.BIT          ;CLEAR PRESENT BIT
025526  005037  002322                        CLR     M1.BIT          ;CLEAR PREVIOUS BIT
025532  012737  022040  002244                MOV     #DMD!MEWD!ECCW,E.MR1 ;SET UP EXPECTED MR1

025540  004737  033414              3$:       JSR     PC,RDBIT        ;GO READ A BIT
025544  016237  000026  002204              MOV     RKMR1(R2),T.MR1 ;GET MR1
025552  023737  002204  002244              CMP     T.MR1,E.MR1     ;CHECK IF IT IS CORRECT
025560  001424                              BEQ     2$              ;YES - SKIP

025562  012737  025632  001236              MOV     #2$,$ESCAPE     ;SET ESCAPE FOR CONTINUE TEST

025570  005037  001236              4$:       CLR     $ESCAPE         ;CLEAR ESCAPE
025574  032777  001000  153336              BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
025602  001011                              BNE     1$              ;YES SKIP
025604  013700  001254                        MOV     $TESTN,R0       ;ELSE SET UP TO SKIP TO NEXT TEST
025610  006300                              ASL     R0
025612  016037  035270  001236              MOV     $SW08TB(R0),$ESCAPE
025620  162737  000002  001236              SUB     #2,$ESCAPE
025626                                  1$:
025626  012600                              MOV     (SP)+,R0        ;;POP STACK INTO R0
025630  000204                              RTS     R4

025632  005237  002326              2$:       INC     BITCNT          ;INCREMENT THE BIT COUNT
025636  005300                              DEC     R0              ;LOOP UNTIL READY FOR READ GATE
025640  001337                              BNE     3$

025642  012700  000177                        MOV     #127.,R0        ;SET COUNT FOR REST OF SYNC
025646  012737  122040  002244                MOV     #DMD!MEWD!ECCW!RDGATE,E.MR1 ;SET NEW EXPECTED MR1

025654  004737  033414              5$:       JSR     PC,RDBIT        ;GO READ BIT
025660  016237  000026  002204              MOV     RKMR1(R2),T.MR1 ;GET MR1
025666  023737  002204  002244              CMP     T.MR1,E.MR1     ;CHECK IF CORRECT
025674  001404                              BEQ     6$              ;YES - SKIP
025676  012737  025706  001236              MOV     #6$,$ESCAPE     ;SET FOR CONTINUE TESTING
025704  000445                              BR      50$             ;GO TO ERROR EXIT
025706  005237  002326              6$:       INC     BITCNT          ;BUMP BIT COUNTER
025712  005300                              DEC     R0              ;LOOP UNTIL READY FOR SYNC 1
025714  001357                              BNE     5$

025716  013737  002320  002322              MOV     PR.BIT,M1.BIT   ;MOV PRESENT TO PREVIOUS
025724  012737  000001  002320              MOV     #1,PR.BIT       ;SET SYNC 1 BIT
025732  004737  033414                      JSR     PC,RDBIT        ;GO READ BIT
025736  016237  000026  002204              MOV     RKMR1(R2),T.MR1 ;GET MR1
025744  023737  002204  002244              CMP     T.MR1,E.MR1     ;CHECK IF CORRECT
025752  001404                              BEQ     7$              ;YES - SKIP
```

```
5949  025754  012737  025764  001236          MOV     #7$,$ESCAPE     ;SET RETURN FOR CONTINUE TEST
5950  025762  000416                          BR      50$             ;GO TO ERROR EXIT
5951
5952  025764  004437  034154          7$:     JSR     R4,GETREG       ;GO GET RK611 REGISTERS
5953  025770  005237  002326                  INC     BITCNT          ;BUMP BIT COUNTER
5954  025774  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR CALL
5955  025776  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
5956  026004  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
5957  026012  001266                          BNE     4$              ;NO - SKIP
5958
5959  026014  005724                          TST     (R4)+           ;BUMP TO NO ERROR RETURN
5960  026016  000703                          BR      1$              ;SKIP TO RETURN
5961
5962  026020  032777  001000  153112  50$:     BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
5963  026026  001403                          BEQ     51$             ;NO - SKIP
5964  026030  005037  001236                  CLR     $ESCAPE         ;CLEAR FOR SKIP TO NEXT TEST
5965  026034  000674                          BR      1$
5966  026036  010446                  51$:    MOV     R4,-(SP)        ;SET UP STACK FOR RETURN
5967  026040  000204                          RTS     R4
5968
5969                                  .SBTTL  DIAGNOSTIC MODE HEADER READ AND COMPARE
5970                                  ;*      THIS ROUTINE SIMULATES THE THREE HEADER WORDS READ AND
5971                                  ;*      COMPARE THAT IS PART OF ALL DATA TRANSFER OPERATIONS.
5972                                  ;*
5973                                  ;*      THE EXPECTED HEADER IS CALCULATED FROM THE CYLINDER, TRACK,
5974                                  ;*      AND SECTOR SPECIFIED IN THE "L" REGISTERS. THE READING OF THE
5975                                  ;*      3 WORDS IS DONE AND MR1 IS MONITORED TO INSURE IT DOES
5976                                  ;*      NOT CHANGE. AT THE END OF THE READ/COMPARE, THE CONTENTS
5977                                  ;*      OF RKCS1 IS CHECKED AND THE INCREMENT OF RKDCYL AND RKDA IS
5978                                  ;*      VERIFIED. IF THE INCREMENT IS GOOD, THE "L" REGISTERS ARE
5979                                  ;*      UPDATED TO THE NEW PACK ADDRESS IN CASE OF MULTI-SECTOR
5980                                  ;*      OPERATIONS. THE BAD SECTOR ERROR DESIRED (BSEDES) SWITCH
5981                                  ;*      IS TESTED AND IF IT IS NON-ZERO, THE 2ND AND 3RD HEADER
5982                                  ;*      WORDS ARE CHANGED ACCORDING TO BSEDES CONTENTS. THIS
5983                                  ;*      CAN BE USED TO FORCE HEADER TYPE ERROR.
5984                                  ;*
5985                                  ;*      CALL:
5986                                  ;*      JSR     R4,DHDCMP
5987                                  ;*      ERROR   12      ;ERROR IN MR1
5988                                  ;*      ERROR   13      ;ERROR IN CS1
5989                                  ;*      ERROR   14      ;ERROR IN PACK ADDRESS
5990
5991  026042                          DHDCMP:
5992  026042  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
5993  026044  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
5994  026046  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5995  026050  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
5996  026052  005037  002326                  CLR     BITCNT          ;CLEAR BIT COUNTER
5997  026056  012700  000005                  MOV     #5,R0           ;SET A SHIFT COUNT
5998  026062  113701  002267                  MOVB    L.DT,R1         ;GET DESIRED TRACK
5999  026066  042701  177400                  BIC     #177400,R1      ;CLEAR UNUSED BITS
6000
6001  026072  006301                  1$:     ASL     R1              ;SHIFT TRACK TO ALIGN TO
6002  026074  005300                          DEC     R0              ;HEADER WORD FORMAT
6003  026076  001375                          BNE     1$
6004
```

```
6005  026100  153701  002266                  BISB    L.DS,R1          ;INSERT DESIRED SECTOR
6006  026104  052701  140000                  BIS     #BIT15!BIT14,R1  ;SET BS BITS
6007  026110  013700  002274                  MOV     L.DCYL,R0        ;GET DESIRED CYLINDER
6008  026114  032737  010000  002260          BIT     #CFMT,L.CS1      ;TEST IF 22 SECTOR FORMAT
6009  026122  001402                          BEQ     2$               ;NO - SKIP
6010  026124  052701  001000                  BIS     #BIT9,R1         ;ELSE SET FORMAT BIT
6011  026130  010003                2$:       MOV     R0,R3            ;COMPUTE VRC
6012  026132  010105                          MOV     R1,R5
6013  026134  040005                          BIC     R0,R5
6014  026136  040103                          BIC     R1,R3
6015  026140  050503                          BIS     R5,R3
6016  026142  005737  002352                  TST     BSEDES           ;TEST IF BSE DESIRED
6017  026146  001404                          BEQ     23$              ;NO - SKIP
6018  026150  043701  002352                  BIC     BSEDES,R1        ;CLEAR BIT IN WORD 2
6019  026154  043703  002352                  BIC     BSEDES,R3        ;CORRECT HVRC
6020  026160                        23$:
6021
6022                                  ;       R0, R1, AND R3 HAVE THE EXPECTED HEADERS
6023
6024  026160  005037  001204                  CLR     $TMP1            ;CLEAR A PASS COUNTER
6025  026164  012737  122040  002244          MOV     #DMD!MEWD!ECCW!RDGATE,E.MR1  ;SET EXPECTED MR1
6026
6027  026172  012705  000001        11$:      MOV     #BIT0,R5         ;ESTABLISH BEGINNING OF DATA WORD
6028  026176  013737  002320  002322  7$:     MOV     PR.BIT,M1.BIT    ;STORE PREVIOUS BIT
6029  026204  005037  002320                  CLR     PR.BIT           ;PRESET FOR PRESENT BIT 0
6030  026210  030500                          BIT     R5,R0            ;TEST IF BIT TO BE A ONE
6031  026212  001403                          BEQ     3$               ;NO - SKIP
6032  026214  012737  000001  002320          MOV     #BIT0,PR.BIT     ;ELSE CHANGE TO ONE
6033  026222  004737  033414                3$:       JSR     PC,RDBIT         ;GO READ A BIT
6034  026226  016237  000026  002204          MOV     RKMR1(R2),T.MR1  ;GET MR1 FOR TESTING
6035  026234  023737  002204  002244          CMP     T.MR1,E.MR1      ;CHECK IF CORRECT
6036  026242  001431                          BEQ     6$               ;YES - SKIP
6037  026244  012737  026326  001236          MOV     #6$,$ESCAPE      ;SET RETURN TO CONTINUE TEST
6038  026252  000137  026756                  JMP     50$
6039
6040  026256  005037  001236        4$:       CLR     $ESCAPE          ;CLEAR TO JUMP TO NEXT TEST
6041  026262  032777  001000  152650          BIT     #BIT9,@SWR       ;CHECK IF LOOP ON ERROR
6042  026270  001011                          BNE     5$               ;YES - SKIP
6043  026272  013705  001254                  MOV     $TESTN,R5        ;GET CURRENT TEST NUMBER
6044  026276  006305                          ASL     R5               ;AND SET UP TO ESCAPE TO NEXT TEST
6045  026300  016537  035270  001236          MOV     $SW08TB(R5),$ESCAPE
6046  026306  162737  000002  001236          SUB     #2,$ESCAPE
6047
6048  026314                        5$:
6049  026314  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
6050  026316  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
6051  026320  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
6052  026322  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
6053  026324  000204                          RTS     R4
6054
6055  026326  005237  002326        6$:       INC     BITCNT           ;BUMP BIT COUNT
6056  026332  005705                          TST     R5               ;CHECK IF LAST BIT OF THIS WORD
6057  026334  100402                          BMI     8$               ;HAS BEEN SIMULATED - YES - SKIP
6058  026336  006305                          ASL     R5               ;SHIFT TO NEXT BIT
6059  026340  000716                          BR      7$               ;GO DO THIS BIT
6060
```

```
6061  026342  005737  001204          8$:     TST     $TMP1           ;TEST IF PASS FOR 1ST WORD JUST DONE
6062  026346  001004                          BNE     9$              ;NO - SKIP
6063  026350  010100                          MOV     R1,R0           ;PUT WORD 2 IN R0 FOR SIMULATION
6064  026352  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
6065  026356  000705                          BR      11$
6066
6067  026360  023727  001204  000002  9$:     CMP     $TMP1,#2        ;JUST DONE WITH WORD 2?
6068  026366  002004                          BGE     10$             ;NO SKIP
6069  026370  010300                          MOV     R3,R0           ;PUT 3RD WORD IN R0
6070  026372  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
6071  026376  000675                          BR      11$
6072
6073  026400  023727  001204  000003  10$:    CMP     $TMP1,#3        ;JUST DONE WITH WORD 3?
6074  026406  002006                          BGE     16$             ;NO - SKIP
6075  026410  005000                          CLR     R0              ;PUT IN WORD 0 OF GAP
6076  026412  012705  100000                  MOV     #BIT15,R5       ;SET TO GIVE IT 1ST GAP BIT
6077  026416  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
6078  026422  000665                          BR      7$
6079
6080  026424  023727  001204  000004  16$:    CMP     $TMP1,#4        ;TEST IF READY FOR GAP
6081  026432  001445                          BEQ     12$             ;YES - SKIP
6082
6083                                  ;       THE FOLLOWING BLOCK OF CODE COMPUTES THE NUMBER OF CLOCK
6084                                  ;       PULSES (WHICH ARE ACTUALLY CALLS TO RDBIT) THE OPERATION
6085                                  ;       BEING PERFORMED REQUIRES TO UPDATE THE ADDRESS COUNTERS
6086                                  ;       (RKDC AND RKDA). THIS CALCULATION IS BASED ON THE WHAT WAS
6087                                  ;       IN THESE COUNTERS WHEN THE OPERATION STARTED (L.DC AND L.DA).
6088
6089  026434  012701  000025                  MOV     #25,R1          ;PRESET R1 FOR 24 SECTOR MODE
6090  026440  032737  010000  002260          BIT     #CFMT,L.CS1     ;IS IT 24 SECTOR MODE OPERATION?
6091  026446  001402                          BEQ     22$             ;YES - SKIP
6092  026450  012701  000023                  MOV     #23,R1          ;ELSE CHANGE FOR 22 SECTOR MODE
6093  026454  032737  002000  002260  22$:    BIT     #CDT,L.CS1      ;RK06?
6094  026462  001403                          BEQ     21$             ;YES - SKIP
6095  026464  052701  002000                  BIS     #BIT10,R1       ;ELSE SET FOR MAX HEAD OF 4
6096  026470  000402                          BR      20$             ;SKIP
6097  026472  052701  001000          21$:    BIS     #BIT9,R1        ;SET FOR MAX HEAD OF 2
6098  026476  023701  002266          20$:    CMP     L.DS,R1         ;TEST IF TRACK/SECTOR ADDRESS WAS MAX
6099  026502  001003                          BNE     17$             ;NO - SKIP
6100  026504  012705  000100                  MOV     #BIT6,R5        ;SET FOR 8 BITS OF GAP TO UPDATE
6101  026510  000410                          BR      19$
6102  026512  123701  002266          17$:    CMPB    L.DS,R1         ;TEST IF SECTOR WAS MAX
6103  026516  001003                          BNE     18$             ;NO - SKIP
6104  026520  012705  000400                  MOV     #BIT8,R5        ;SET FOR 6 BITS OF GAP TO UPDATE
6105  026524  000402                          BR      19$
6106  026526  012705  010000          18$:    MOV     #BIT12,R5       ;SET TO GIVE 1ST 4 BITS OF GAP
6107  026532  005237  001204          19$:    INC     $TMP1           ;BUMP PASS COUNT
6108  026536  042737  100000  002244          BIC     #RDGATE,E.MR1   ;CLEAR READ GATE
6109  026544  000614                          BR      7$
6110                                  ;       END OF BLOCK
6111
6112  026546  004437  034154          12$:    JSR     R4,GETREG       ;GET RK611 REGISTERS
6113  026552  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6114  026554  013737  002260  002220          MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
6115  026562  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
6116  026570  001232                          BNE     4$              ;NO SKIP
```

```
6117
6118  026572  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6119  026574  013737  002274  002240          MOV     L.DCYL,E.DCYL   ;GET STARTING CYLINDER
6120  026602  013737  002266  002226          MOV     L.DA,E.DA       ;                TRACK/SECTOR
6121  026610  105237  002226                  INCB    E.DA            ;BUMP SECTOR
6122  026614  012700  000026                  MOV     #26,R0          ;PRESET FOR 26 SEC/TRACK
6123  026620  032737  010000  002260          BIT     #CFMT,L.CS1     ;TEST IF IN 26 SECTOR MODE
6124  026626  001402                          BEQ     13$             ;YES - SKIP
6125  026630  012700  000024                  MOV     #24,R0          ;ELSE CHANGE FOR 24 SECTOR MODE
6126  026634  120037  002226          13$:    CMPB    R0,E.DA         ;CHECK IF SECTOR TO LARGE
6127  026640  001023                          BNE     15$             ;NO - SKIP
6128  026642  105037  002226                  CLRB    E.DA            ;SET SECTOR TO 0
6129  026646  105237  002227                  INCB    E.DA+1          ;BUMP TRACK
6130  026652  012700  000003                  MOV     #3,R0           ;PRESET FOR 3 TRACKS (RK06)
6131  026656  032737  002000  002260          BIT     #CDT,L.CS1      ;TEST IF IT IS RK06
6132  026664  001402                          BEQ     14$             ;YES - SKIP
6133  026666  012700  000005                  MOV     #5,R0           ;ELSE CHANGE
6134  026672  123700  002227          14$:    CMPB    E.DA+1,R0       ;CHECK IF TRACK INC TO LARGE
6135  026676  001004                          BNE     15$             ;NO - SKIP
6136
6137  026700  105037  002227                  CLRB    E.DA+1          ;CLEAR TRACK TO 0
6138  026704  005237  002240                  INC     E.DCYL          ;BUMP CYLINDER
6139
6140  026710  023737  002240  002200  15$:    CMP     E.DCYL,T.DCYL   ;CHECK IF CYL OK
6141  026716  001004                          BNE     25$             ;NO - SKIP
6142  026720  023737  002226  002166          CMP     E.DA,T.DA       ;CHECK IF DA OK
6143  026726  001402                          BEQ     26$
6144  026730  000137  026256          25$:    JMP     4$
6145
6146  026734  013737  002240  002274  26$:    MOV     E.DCYL,L.DCYL   ;STORE COMPUTED NEXT CYL
6147  026742  013737  002226  002266          MOV     E.DA,L.DA       ;AND NEXT DA
6148
6149  026750  005724                          TST     (R4)+           ;BUMP TO NO ERROR RETURN
6150  026752  000137  026314                  JMP     5$              ;GO TO GOOD EXIT
6151
6152  026756  032777  001000  152154  50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
6153  026764  001404                          BEQ     51$             ;NO - SKIP
6154  026766  005037  001236                  CLR     $ESCAPE         ;CLEAR TO LOOP ON ERROR
6155  026772  000137  026314                  JMP     5$
6156
6157  026776  010446                  51$:    MOV     R4,-(SP)        ;SET STACK TO CONTINUE TEST
6158  027000  000204                          RTS     R4
6159                                  .SBTTL  DIAGNOSTIC MODE GAP READ AND SYNC WRITE
6160                                  ;*      THE READING OF THE GAP AND WRITING OF THE SYNC (OR PREAMBLE)
6161                                  ;*      IS HANDLED IN THIS ROUTINE.
6162                                  ;*
6163                                  ;*      THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
6164                                  ;*      ROUTINE AND THE REMAINDER ARE PROCESSED HERE. THEN A
6165                                  ;*      CHECK IS MADE THAT WRITE GATE CAME ON AND 255 "0" BITS
6166                                  ;*      AND A SINGLE "1" BIT IS WRITTEN.
6167                                  ;*
6168                                  ;*      CALL:
6169                                  ;*      JSR     R4,DWGPSN
6170                                  ;*      ERROR   15      ;MR1 IN ERROR IN READ GAP
6171                                  ;*      ERROR   16      ;CS1 IN ERROR AFTER READ GAP
6172                                  ;*      ERROR   17      ;MR1 IN ERROR IN WRITE SYNC
```

```
                                        ;*      ERROR   20      ;CS1 IN ERROR AFTER SYNC WRITE
                                        ;*
                                        ;*      ROUTINE CALLED:
                                        ;*      JSR     PC,WRTBIT
                                        ;*      RETURN POINT IF ERROR IN WRTBIT
                                        ;*      NO ERROR RETURN

027002                                  DWGPSN:
027002  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        ;       THE FOLLOWING CODE ADJUST THE COUNT FOR THE REMAINDER OF
                                        ;       THE GAP. THE ADJUSTMENT DEPENDS ON HOW MANY GAP BITS WERE
                                        ;       READ IN THE HEADER COMPARE ROUTINE FOR ADDRESS REGISTER UPDATE.
027004  005737  002266                        TST     L.DA            ;TEST IF DESIRED TRACK/SECTOR NOW 0
027010  001006                                BNE     8$              ;NO - SKIP
027012  012700  000066                        MOV     #54.,R0         ;ELSE ADJUST COUNT FOR CYL CHANGE
027016  012737  000011  002326                MOV     #11,BITCNT      ;PRESET BIT COUNT FOR REST OF GAP
027024  000416                                BR      10$
027026  105737  002266              8$:       TSTB    L.DS            ;TEST IF NEW SECTOR IS 0
027032  001006                                BNE     9$              ;NO - SKIP
027034  012700  000070                        MOV     #56.,R0         ;ELSE ADJUST COUNT FOR TRACK CHANGE
027040  012737  000007  002326                MOV     #7,BITCNT       ;PRESET BIT COUNT FOR REST OF GAP
027046  000405                                BR      10$
027050  012737  000005  002326        9$:     MOV     #5,BITCNT       ;PRESET BIT COUNTER FOR REST OF GAP
027056  012700  000074                        MOV     #60.,R0         ;SET COUNT FOR 60 BITS
                                        ;       END OF BLOCK

027062  012737  022040  002244        10$:    MOV     #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
027070  005037  002320                        CLR     PR.BIT          ;CLEAR PRESENT BIT
027074  005037  002322                        CLR     M1.BIT          ;CLEAR PREVIOUS BIT
027100  004737  033414              1$:       JSR     PC,RDBIT        ;GO READ BIT
027104  016237  000026  002204                MOV     RKMR1(R2),T.MR1 ;GET MR1
027112  023737  002244  002204                CMP     E.MR1,T.MR1     ;CHECK IF CORRECT
027120  001404                                BEQ     11$             ;YES - SKIP
027122  012737  027144  001236                MOV     #2$,$ESCAPE     ;SET RETURN TO CONT TEST
027130  000513                                BR      50$
027132  005237  002326              11$:      INC     BITCNT          ;BUMP BIT COUNT
027136  005300                                DEC     R0              ;DEC LOOP COUNT
027140  001357                                BNE     1$              ;LOOP IF NOT ZERO
027142  000421                                BR      4$              ;ELSE SKIP

027144  005037  001236              2$:       CLR     $ESCAPE         ;CLEAR TO EXIT OR LOOP
027150  022777  001000  151762                CMP     #BIT9,@SWR      ;ERROR RETURN: CHECK IF LOOP ON ERROR
027156  001011                                BNE     3$              ;YES - SKIP
027160  013700  001254                        MOV     $TESTN,R0       ;GET NUMBER OF NEXT TEST
027164  006300                                ASL     R0              ;SHIFT FOR INDEXING
027166  016037  035270  001236                MOV     $SWOBTB(R0),$ESCAPE  ;SET ESCAPE FOR GOT TO NEXT TEST
027174  162737  000002  001236                SUB     #2,$ESCAPE      ;SET TO FIRST INST (SCOPE)
027202                                  3$:
027202  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
027204  000204                                RTS     R4

027206  004437  034154              4$:       JSR     R4,GETREG       ;GET RK611 REGS
027212  005724                                TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
027214  013737  002260  002220                MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
027222  023737  002160  002220                CMP     T.CS1,E.CS1     ;CHECK IF CS1 OK
027230  001345                                BNE     2$              ;NO - SKIP
```

```
6229
6230  027232  005724                          TST     (R4)+           ;BUMP RETURN POINTER
6231
6232  027234  005037  002326                  CLR     BITCNT          ;CLEAR BIT COUNTER
6233  027240  005037  002322                  CLR     M1.BIT          ;CLEAR ALL WRITE BIT INDICATORS
6234  027244  005037  002324                  CLR     M2.BIT
6235  027250  005037  002316                  CLR     P1.BIT
6236  027254  005037  002320                  CLR     PR.BIT
6237  027260  012737  062040  002244          MOV     #DMD!ECCW!MEWD!WRTGATE,E.MR1 ;SET EXPECTED MR1
6238  027266  012700  000400                  MOV     #256.,R0        ;SET COUNT TO WRITE 256 "0"
6239
6240  027272  004737  032450          5$:     JSR     PC,WRTBIT       ;CALL WRITE BIT
6241  027276  000405                          BR      6$              ;ERROR RETURN -'SKIP TO HANDLER
6242  027300  005237  002326                  INC     BITCNT          ;BUMP BIT COUNT
6243  027304  005300                          DEC     R0              ;LOOP UNTIL ALL 256
6244  027306  001371                          BNE     5$              ;BITS ARE WRITTEN
6245  027310  000402                          BR      7$
6246
6247  027312  010446                  6$:     MOV     R4,-(SP)        ;WRITE ERROR HANDLER-SET
6248  027314  000204                          RTS     R4              ;STACK FOR REPORTING SUBSEQUENT ERRORS
6249
6250  027316  012737  000001  002316  7$:     MOV     #1,P1.BIT       ;SET SYNC BIT OF 1 AS NEXT BIT
6251  027324  004737  032450                  JSR     PC,WRTBIT       ;GO WRITE IT
6252  027330  000770                          BR      6$              ;ERROR RETURN
6253  027332  005237  002326                  INC     BITCNT          ;BUMP BIT COUNT
6254
6255  027336  004437  034154                  JSR     R4,GETREG       ;GET RK611 REGS
6256  027342  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6257  027344  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
6258  027352  001274                          BNE     2$              ;NO - SKIP
6259
6260  027354  005724                          TST     (R4)+           ;BUMP PAST ERROR RETURNS
6261  027356  000711                          BR      3$              ;GO TO EXIT
6262
6263  027360  032777  001000  151552  50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
6264  027366  001403                          BEQ     51$             ;YES - SKIP
6265  027370  005037  001236                  CLR     $ESCAPE         ;ELSE SET TO LOOP
6266  027374  000702                          BR      3$
6267  027376  010446                  51$:    MOV     R4,-(SP)        ;PRESET STACK FOR RETURN
6268  027400  000204                          RTS     R4
6269                                  .SBTTL  DIAGNOSTIC WRITE DATA AND ECC ROUTINE
6270                                  ;*      THE DATA IN THE OUPUT BUFFER IS WRITE SIMULATED, ECC IS
6271                                  ;*      GENERATED AND WRITTEN, AND THE POSTAMBLE IS WRITTEN.
6272                                  ;*
6273                                  ;*      CALL:
6274                                  ;*      JSR     R4,DWRITE
6275                                  ;*      ERROR   17      ;MR1 IN ERROR IN WRITING
6276                                  ;*      ERROR   21      ;ECC ERROR IN WRITING
6277                                  ;*      ERROR   22      ;CS1 ERROR AFTER WRITE
6278                                  ;*      NO ERROR RETURN
6279                                  ;*
6280                                  ;*      ROUTINE CALLED:
6281                                  ;*      JSR     PC,WRTBIT
6282                                  ;*      RETURN POINT IF ERROR IN WRTBIT
6283                                  ;*      NO ERROR RETURN FROM WRTBIT
6284
```

```
                                DWRITE:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                        CLR     BITCNT          ;CLEAR BIT COUNTER
                                        CLR     ECCLO           ;CLEAR ECC LOW
                                        CLR     ECCHI           ;CLEAR ECC HI
                                        MOV     #400,R0         ;SET WORD COUNT FOR ONE SECTOR
                                        MOV     #OBUFF,R3       ;GET POINTER TO OUTPUT DATA
                                        MOV     R4,$TMP6        ;STORE MR1 ERROR RETURN
                                        TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
                                        MOV     R4,$TMP7        ;STORE ECC ERROR RETURN

                                ;       THE FOLLOWING CODE SIMULATES THE WRITING OF 400 WORDS.
                                ;       THE PROCEDURE IS TO LOAD THE CURRENT WORD INTO R5 AND
                                ;       TO SHIFT A BIT FROM RIGHT TO LEFT TO POINT TO THE CURRENT BIT.

                                        CLR     $TMP5           ;CLEAR SWITCH
                                3$:     MOV     #BIT0,R1        ;LOAD BIT POINTER
                                        MOV     (R3)+,R5        ;GET A WORD FOR WRITING

                                4$:     MOV     M1.BIT,M2.BIT   ;SHIFT CURRENT-1 TO CURRENT-2
                                        MOV     PR.BIT,M1.BIT   ;PRESENT TO CURRENT-1
                                        MOV     P1.BIT,PR.BIT   ;NEXT TO PRESENT
                                        CLR     P1.BIT          ;CLEAR NEXT BIT
                                        BIT     R1,R5           ;TEST IF NEXT BIT IS ONE
                                        BEQ     5$              ;NO - SKIP
                                        MOV     #1,P1.BIT       ;ELSE SET NEXT BIT
                                5$:     JSR     PC,WRTBIT       ;GO WRITE BIT
                                        BR      33$             ;ERROR RETURN - SKIP
                                43$:    JSR     PC,ECCGEN       ;GO GENERATE ECC FOR THIS BIT
                                        MOV     RKECPT(R2),T.ECPT ;GET PATTERN
                                        CMP     E.ECPT,T.ECPT   ;CHECK IF ECC CORRECT
                                        BEQ     40$             ;YES - SKIP
                                        JMP     13$             ;JUMP TO ERROR EXIT

                                40$:    INC     BITCNT          ;BUMP BIT COUNT
                                        TST     R1              ;TEST BIT POINTER
                                        BMI     7$              ;IF NEGATIVE - SKIP
                                        ASL     R1              ;ELSE SHIFT LEFT AND LOOP
                                        CMP     R0,#1           ;TEST IF WRITING LAST WORD
                                        BNE     4$              ;NO - GO TO LOOP
                                        BIT     #CFMT,L.CS1     ;TEST IF 22 SECTOR MODE
                                        BEQ     32$             ;NO - SKIP
                                        TST     $TMP5           ;TEST IF 18 BIT SWITCH SET
                                        BNE     32$             ;YES - SKIP
                                        BR      4$              ;ELSE LOOP
                                32$:    TST     R1              ;ELSE TEST IF LAST BIT OF LAST WORD
                                        BPL     4$              ;NO - GO TO LOOP
                                        BIC     #ECCW,E.MR1     ;ELSE SET MR1 FOR ECC WRITE
                                        BR      4$
                                33$:    JMP     20$

                                7$:     BIT     #CFMT,L.CS1     ;TEST IF 22 SECTOR MODE
                                        BEQ     31$             ;NO - SKIP
```

```
6341  027646  005737  001214            TST     $TMP5           ;TEST IF 18 BIT SWITCH SET
6342  027652  001006                    BNE     30$             ;YES - SKIP
6343  027654  012701  040000            MOV     #BIT14,R1       ;SET TO SUPPLY 2 MORE BITS (16 & 17)
6344  027660  005005                    CLR     R5              ;MAKE SURE THEY ARE 0'S
6345  027662  005137  001214            COM     $TMP5           ;SET THE 18 BIT SWITCH
6346  027666  000675                    BR      4$              ;GO WRITE THE BITS
6347
6348  027670  005037  001214    30$:    CLR     $TMP5           ;CLEAR THE SWITCH
6349  027674  005300            31$:    DEC     R0              ;DEC WORD COUNT
6350  027676  001266                    BNE     3$              ;IF NOT ZERO - GO GET NEXT WORD
6351  027700  005037  001214            CLR     $TMP5           ;CLEAR FOR PASS CONTROL
6352  027704  013705  002340            MOV     ECCHI,R5        ;GET FIRST ECC WORD
6353  027710  012701  000001            MOV     #BIT0,R1        ;SET BIT POINTER
6354
6355  027714  013737  002322  002324    MOV     M1.BIT,M2.BIT   ;SHIFT BIT INDICATORS AGAIN
6356  027722  013737  002320  002322    MOV     PR.BIT,M1.BIT
6357  027730  013737  002316  002320    MOV     P1.BIT,PR.BIT
6358  027736  005037  002316            CLR     P1.BIT          ;CLEAR NEXT BIT
6359  027742  030105                    BIT     R1,R5           ;TEST IF BIT IS ZERO
6360  027744  001403                    BEQ     9$              ;YES - SKIP
6361  027746  012737  000001  002316    MOV     #1,P1.BIT       ;ELSE CHANGE TO ONE
6362  027754  004737  032450    9$:     JSR     PC,WRTBIT       ;GO WRITE BIT
6363  027760  000501                    BR      14$             ;WRTBIT ERROR - SKIP
6364
6365  027762  005237  002326    41$:    INC     BITCNT          ;BUMP BIT COUNT
6366  027766  005701                    TST     R1              ;TEST BIT POINTER
6367  027770  100413                    BMI     10$             ;IF NEGATIVE - SKIP
6368  027772  006301                    ASL     R1              ;ELSE SHIFT AND LOOP
6369  027774  023727  001214  000001    CMP     $TMP5,#1        ;TEST IF WRITING LAST ECC WORD
6370  030002  001344                    BNE     8$              ;NO - SKIP TO LOOP
6371  030004  005701                    TST     R1              ;ELSE TEST IF WRITING LAST BIT OF LAST ECC
6372  030006  100342                    BPL     8$              ;NO - SKIP TO LOOP
6373  030010  052737  020000  002244    BIS     #ECCW,E.MR1     ;ELSE SET ECCW IN MR1
6374  030016  000736                    BR      8$
6375
6376  030020  005737  001214    10$:    TST     $TMP5           ;TEST PASS  0 JUST DONE
6377  030024  001007                    BNE     11$             ;NO - SKIP
6378  030026  005237  001214            INC     $TMP5           ;BUMP PASS COUNT
6379  030032  012701  000001            MOV     #BIT0,R1        ;PRESET BIT POINTER
6380  030036  013705  002342            MOV     ECCLO,R5        ;GET SECOND ECC WORD
6381  030042  000724                    BR      8$              ;LOOP
6382
6383  030044  012701  000017    11$:    MOV     #15.,R1         ;SET A BIT COUNT
6384
6385  030050  013737  002322  002324    MOV     M1.BIT,M2.BIT   ;SHIFT BIT INDICATORS
6386  030056  013737  002320  002322    MOV     PR.BIT,M1.BIT
6387  030064  013737  002316  002320    MOV     P1.BIT,PR.BIT
6388  030072  005037  002316            CLR     P1.BIT          ;SET ZERO FOR WRITE POSTAMBLE
6389  030076  004737  032450            JSR     PC,WRTBIT       ;GO WRITE BIT
6390  030102  000436                    BR      15$             ;WRITE BIT ERROR - SKIP
6391  030104  005237  002326    42$:    INC     BITCNT          ;BUMP BIT COUNT
6392  030110  005301                    DEC     R1              ;DEC BIT COUNT
6393  030112  001356                    BNE     12$             ;LOOP UNTTIL BIT COUNT 0
6394
6395  030114  004437  034154            JSR     R4,GETREG       ;GET RK611 REGS
6396  030120  013704  001220            MOV     $TMP7,R4        ;SET R4 FOR ECC ERROR RETURN
```

```
6397  030124  005724                          TST     (R4)+           ;BUMP TO CS1 ERROR RETURN
6398  030126  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
6399
6400  030134  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
6401  030142  001031                          BNE     18$             ;NO - SKIP
6402  030144  005724                          TST     (R4)+           ;BUMP TO GOOD RETURN
6403  030146  000446                          BR      19$             ;GO TO RETURN
6404
6405  030150  013704  001220          13$:    MOV     $TMP7,R4        ;SET FOR ECC ERROR RETURN
6406  030154  012737  027556  001236          MOV     #40$,$ESCAPE    ;STORE RETURN POINT FOR NO LOOP ON ERROR
6407  030162  000410                          BR      16$
6408
6409  030164  013704  001216          14$:    MOV     $TMP6,R4        ;SET FOR MR1 ERROR RETURN
6410  030170  000405                          BR      16$
6411
6412  030172  013704  001216          20$:    MOV     $TMP6,R4        ;SET FOR MR1 ERROR RETURN
6413  030176  000402                          BR      16$
6414
6415  030200  013704  001216          15$:    MOV     $TMP6,R4        ;SET FOR MR1 ERROR RETURN
6416
6417  030204  032777  001000  150726  16$:    BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
6418  030212  001403                          BEQ     17$             ;NO - SKIP
6419
6420  030214  005037  001236                  CLR     $ESCAPE         ;CLEAR ESCAPE
6421  030220  000421                          BR      19$             ;SKIP TO EXIT
6422
6423  030222  010446          17$:    MOV     R4,-(SP)        ;CONDITION STACK
6424  030224  000204                  RTS     R4              ;GO REPORT ERROR
6425
6426  030226  005037  001236          18$:    CLR     $ESCAPE         ;CLEAR FOR LOOP OR EXIT
6427  030232  032777  001000  150700          BIT     #BIT9,@SWR      ;CHECK IF LOOP ON ERROR
6428  030240  001011                          BNE     19$             ;YES - SKIP
6429  030242  013700  001254                  MOV     $TESTN,R0       ;SET UP TO SKIP TO NEXT TEST
6430  030246  006300                          ASL     R0
6431  030250  016037  035270  001236          MOV     $SW08TB(R0),$ESCAPE
6432  030256  162737  000002  001236          SUB     #2,$ESCAPE
6433  030264                          19$:
6434  030264  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
6435  030266  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
6436  030270  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
6437  030272  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
6438  030274  000204                          RTS     R4
6439
6440                          .SBTTL  DIAGNOSTIC MODE READ GAP AND DATA SYNC
6441                          ;*      THE READING OF THE GAP AND READING OF THE SYNC (OR PREAMBLE)
6442                          ;*      IS HANDLED IN THIS ROUTINE.
6443                          ;*
6444                          ;*      THE FIRST GAP BITS ARE PROCESSED IN THE HEADER COMPARE
6445                          ;*      ROUTINE AND THE REMAINING GAP BITS ARE PROCESSED HERE.
6446                          ;*      THEN A CHECK IS MADE THAT READ GATE CAME ON
6447                          ;*      AT THE 129TH BIT OF SYNC.;*
6448                          ;*      CALL:
6449                          ;*      JSR     R4,DRGPSN
6450                          ;*      ERROR   15      ;MR1 IN ERROR IN READ GAP
6451                          ;*      ERROR   16      ;CS1 IN ERROR AFTER READ GAP
6452                          ;*      ERROR   32      ;MR1 IN ERROR IN READ SYNC
```

```
                                        ;*      ERROR   33      ;CS1 IN ERROR AFTER SYNC READ
                                        ;*
                                        ;*      ROUTINE CALLED:
                                        ;*      JSR     PC,RDBIT
                                        DRGPSN:
                                                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        ;       THE FOLLOWING CODE ADJUST THE COUNT FOR THE REMAINDER OF
                                        ;       THE GAP.  THE ADJUSTMENT DEPENDS ON HOW MANY GAP BITS WERE
                                        ;       READ IN THE HEADER COMPARE ROUTINE FOR ADDRESS REGISTER UPDATE.
                                                TST     L.DA            ;TEST IF DESIRED TRACK/SECTOR NOW 0
                                                BNE     8$              ;NO - SKIP
                                                MOV     #54.,R0         ;ELSE ADJUST COUNT FOR CYL CHANGE
                                                MOV     #11,BITCNT      ;PRESET BIT COUNT FOR REST OF GAP
                                                BR      10$
                                        8$:     TSTB    L.DS            ;TEST IF NEW SECTOR IS 0
                                                BNE     9$              ;NO - SKIP
                                                MOV     #56.,R0         ;ELSE ADJUST COUNT FOR TRACK CHANGE
                                                MOV     #7,BITCNT       ;PRESET BIT COUNT FOR REST OF GAP
                                                BR      10$
                                        9$:     MOV     #5,BITCNT       ;PRESET BIT COUNTER FOR REST OF GAP
                                                MOV     #60.,R0         ;SET COUNT FOR 60 BITS
                                        ;       END OF BLOCK

                                        10$:    MOV     #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
                                                CLR     PR.BIT          ;CLEAR PRESENT BIT
                                                CLR     M1.BIT          ;CLEAR PREVIOUS BIT
                                        1$:     JSR     PC,RDBIT        ;GO READ BIT
                                                MOV     RKMR1(R2),T.MR1 ;GET MR1
                                                CMP     E.MR1,T.MR1     ;CHECK IF CORRECT
                                                BEQ     11$             ;YES - SKIP
                                                MOV     #2$,$ESCAPE     ;SET RETURN TO CONT TEST
                                                BR      50$
                                        11$:    INC     BITCNT          ;BUMP BIT COUNT
                                                DEC     R0              ;DEC LOOP COUNT
                                                BNE     1$              ;LOOP IF NOT ZERO
                                                BR      4$              ;ELSE SKIP

                                        2$:     CLR     $ESCAPE         ;CLEAR TO EXIT OR LOOP
                                                CMP     #BIT9,@SWR      ;ERROR RETURN: CHECK IF LOOP ON ERROR
                                                BNE     3$              ;YES - SKIP
                                                MOV     $TESTN,R0       ;GET NUMBER OF NEXT TEST
                                                ASL     R0              ;SHIFT FOR INDEXING
                                                MOV     $SW08TB(R0),$ESCAPE  ;SET ESCAPE FOR GOT TO NEXT TEST
                                                SUB     #2,$ESCAPE      ;SET TO FIRST INST (SCOPE)
                                        3$:
                                                MOV     (SP)+,R0        ;;POP STACK INTO R0
                                                RTS     R4

                                        4$:     JSR     R4,GETREG       ;GET RK611 REGS
                                                TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
                                                MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
                                                CMP     T.CS1,E.CS1     ;CHECK IF CS1 OK
                                                BNE     2$              ;NO - SKIP

                                                TST     (R4)+           ;BUMP RETURN POINTER
```

```
6509
6510  030530  005037  002326                  CLR     BITCNT          ;CLEAR BIT COUNTER
6511  030534  012737  022040  002244          MOV     #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
6512  030542  012700  000200                  MOV     #128.,R0        ;SET COUNT TO READ 128 "0"
6513
6514  030546  004737  033414          5$:     JSR     PC,RDBIT        ;CALL READ BIT
6515  030552  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;GET MR1 FOR TESTING
6516  030560  023737  002244  002204          CMP     E.MR1,T.MR1     ;TEST IF CORRECT
6517  030566  001404                          BEQ     22$             ;YES - SKIP
6518
6519  030570  012737  030600  001236          MOV     #22$,$ESCAPE    ;SET RETURN FOR CONTINUE TEST
6520  030576  000465                          BR      50$             ;GO TO ERROR EXIT
6521  030600  005237  002326          22$:    INC     BITCNT          ;BUMP BIT COUNT
6522  030604  005300                          DEC     R0              ;LOOP UNTIL ALL 128
6523  030606  001357                          BNE     5$              ;BITS ARE READ
6524  030610  012700  000177                  MOV     #127.,R0        ;SET COUNT FOR REST OF SYNC
6525  030614  052737  100000  002244          BIS     #RDGATE,E.MR1   ;SET EXPECTED MR1
6526  030622  004737  033414          6$:     JSR     PC,RDBIT        ;GO CALL READ BIT
6527  030626  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;GET MR1 FOR TESTING
6528  030634  023737  002244  002204          CMP     E.MR1,T.MR1     ;TEST IF CORRECT
6529  030642  001404                          BEQ     23$             ;YES - SKIP
6530  030644  012737  030654  001236          MOV     #23$,$ESCAPE    ;ELSE SET ESCAPE
6531  030652  000437                          BR      50$             ;GO TO ERROR REPORT EXIT
6532
6533  030654  005237  002326          23$:    INC     BITCNT          ;BUMP BIT COUNTER
6534  030660  005300                          DEC     R0              ;DEC BIT COUNT
6535  030662  001357                          BNE     6$              ;LOOP UNTIL ALL 127 READ
6536
6537  030664  012737  000001  002320          MOV     #1,PR.BIT       ;SET SYNC BIT OF 1 AS NEXT BIT
6538  030672  004737  033414                  JSR     PC,RDBIT        ;GO READ IT
6539  030676  016237  000026  002244          MOV     RKMR1(R2),E.MR1 ;GET MR1 FOR TESTING
6540  030704  023737  002244  002204          CMP     E.MR1,T.MR1     ;CHECK IF CORRECT
6541  030712  001404                          BEQ     24$             ;YES - SKIP
6542  030714  012737  030724  001236          MOV     #24$,$ESCAPE    ;ELSE SET RETURN FOR CONTINUE TEST
6543  030722  000413                          BR      50$
6544
6545  030724  005237  002326          24$:    INC     BITCNT          ;BUMP BIT COUNT
6546
6547  030730  004437  034154                  JSR     R4,GETREG       ;GET RK611 REGS
6548  030734  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6549  030736  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
6550  030744  001235                          BNE     2$              ;NO - SKIP
6551
6552  030746  005724                          TST     (R4)+           ;BUMP PAST ERROR RETURNS
6553  030750  000652                          BR      3$              ;GO TO EXIT
6554
6555  030752  032777  001000  150160  50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
6556  030760  001403                          BEQ     51$             ;YES - SKIP
6557  030762  005037  001236                  CLR     $ESCAPE         ;ELSE SET TO LOOP
6558  030766  000643                          BR      3$
6559  030770  010446                  51$:    MOV     R4,-(SP)        ;PRESET STACK FOR RETURN
6560  030772  000204                          RTS     R4
6561                                  .SBTTL  DIAGNOSTIC READ DATA OR DIAGNOSTIC WRITE CHECK ROUTINE
6562                                  ;*      THIS ROUTINE IS USED TO SIMULATE THE READ OR WRITE CHECK
6563                                  ;*      OF DATA. THE ROUTINE WILL TRANSFER A FULL OR PARTIAL SECTOR.
6564                                  ;*      THE FIRST WORD AFTER THE CALL SPECIFIES HOW MANY WORDS
```

```
                                ;*      ARE TO BE TRANSFERRED.
                                ;*
                                ;*      IF LESS THAN A FULL SECTOR, THE ECC AT THE END
                                ;*      OF THE TRANSFER IS NOT CHECKED. THE ECC IS CHECKED AT THE TIME
                                ;*      EACH BIT IS TRANSFERED.
                                ;*
                                ;*      EITHER 16 OR 18 BIT WORDS CAN BE SIMULATED
                                ;*      DEPENDING ON THE STATE OF CFMT BIT IN L.CS1. IF 18 BITS
                                ;*      ARE REQUIRED, BITS 16 AND 17 ARE ASSUMED TO BE ZERO.
                                ;*
                                ;*      THE CONTENTS OF LOCATION ECPOSX MUST BE LOADED WITH THE ECC
                                ;*      COUNT EXPECTED AT THE END OF THE DATA TRANSFER. THIS
                                ;*      LOADING MUST OCCUR BEFORE THE ROUTINE IS CALLED.
                                ;*
                                ;*      ANOTHER LOCATION, ECPATX, IS SET TO 0 IF THE ECC IS EXPECTED
                                ;*      TO BE CORRECT AND SET TO 1 IF AN ERROR IS BEING FORCED.
                                ;*
                                ;*      A 3RD LOCATION, ECCSRC, SPECIFIES THE SOURCE OF THE ECC
                                ;*      WORDS TO BE USED IN CHECKING THE OPERATION. IF ECCSRC IS A
                                ;*      1, THE ECC WORDS FED INTO THE DECODER ARE TAKEN FROM THE
                                ;*      LAST TWO LOCATIONS OF OBUFF, IF ECCSRC IS 0, THE COMPUTED
                                ;*      ECC WORDS FROM ECCHI AND ECCLO ARE USED.
                                ;*      CALL:
                                ;*      JSR     R4,DREAD
                                ;*      LENGTH OF TRANSFER
                                ;*      ERROR 34        ;MR1 IN ERROR READING DATA OR ECC
                                ;*      ERROR 35        ;ECC ERROR READING DATA
                                ;*      ERROR 36        ;CS1 ERROR AFTER READING DATA OR ECC
                                ;*      ERROR 41        ;ECC REGISTER INCORRECT AFTER READ ECC
                                ;*      ERROR 42        ;ERR IN ECC PAT CALCULATION
                                ;*      ERROR 43        ;ERR IN ECC POS COUNTING
                                ;*
                                ;*      OR
                                ;*
                                ;*      JSR     R4,DWRTCK
                                ;*      LENGTH OF TRANSFER
                                ;*      ERROR 53        ;MR1 IN ERROR WRITE CHK OR ECC READ
                                ;*      ERROR 54        ;ECC ERROR IN WRITE CHECK
                                ;*      ERROR 55        ;CS1 ERROR AFTER IN WRT CHK OR ECC READ
                                ;*      ERROR 41        ;ECC REGISTER INCORRECT AFTER READ ECC
                                ;*      ERROR 42        ;ERR IN ECC PAT CALCULATION
                                ;*      ERROR 43        ;ERR IN ECC POS COUNTING
                                ;*
                                ;*      ROUTINES CALLED:
                                ;*      RDBIT
                                ;*      ECCGEN
                                ;*
030774                          DWRTCK:
030774                          DREAD:
030774  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
030776  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
031000  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
031002  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
031004  012400                          MOV     (R4)+,R0        ;STORE WORD COUNT
031006  010437  001216                  MOV     R4,$TMP6        ;STORE MR1 ERROR RETURN
```

```
6621  031012  005724                        TST     (R4)+           ;BUMP TO NEXT RETURN
6622  031014  010437  001220                MOV     R4,$TMP7        ;STORE ECC ERROR RETURN
6623  031020  005724                        TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6624  031022  010437  001210                MOV     R4,$TMP3        ;STORE CS1 ERROR RETURN
6625  031026  005724                        TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6626  031030  010437  001212                MOV     R4,$TMP4        ;STORE ECC REG ERROR RETURN
6627  031034  005724                        TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6628  031036  010437  001222                MOV     R4,$TMP10       ;STORE ECC PAT ERR RETURN
6629  031042  005724                        TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
6630  031044  010437  001224                MOV     R4,$TMP11       ;SOTER ECC POS ERROR RETURN
6631  031050  012703  051444                MOV     #OBUFF,R3       ;GET ADDRESS OF BUFFER
6632  031054  005037  002326                CLR     BITCNT          ;CLEAR BIT COUNTER
6633  031060  005037  002340                CLR     ECCHI           ;CLEAR ECC COMPUTED LOCATIONS
6634  031064  005037  002342                CLR     ECCLO
6635  031070  005037  001214                CLR     $TMP5           ;CLEAR FOR 18 BIT MODE SWITCH
6636  031074  032737  010000  002260        BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
6637  031102  001412                        BEQ     32$             ;NO - SKIP
6638  031104  012737  011000  001226        MOV     #11000,$TMP12   ;ELSE SET UP BIT COUNTS FOR 18 BIT MODE
6639  031112  012737  011001  001230        MOV     #11001,$TMP13
6640  031120  012737  011041  001232        MOV     #11041,$TMP14
6641  031126  000411                        BR      3$
6642  031130  012737  010000  001226  32$:   MOV     #10000,$TMP12   ;SET BIT COUNTS FOR 16 BIT MODE
6643  031136  012737  010001  001230        MOV     #10001,$TMP13
6644  031144  012737  010041  001232        MOV     #10041,$TMP14
6645
6646  031152  012701  000001        3$:     MOV     #BIT0,R1        ;PRESET BIT POINTER
6647  031156  012305                        MOV     (R3)+,R5        ;GET DATA WORD
6648
6649  031160  013737  002320  002322  4$:   MOV     PR.BIT,M1.BIT   ;SHIFT PRESENT INTO PREVIOUS
6650  031166  005037  002320                CLR     PR.BIT          ;CLEAR PRESENT
6651  031172  030105                        BIT     R1,R5           ;TEST IF PRESENT BIT IS 0
6652  031174  001403                        BEQ     1$              ;YES - SKIP
6653  031176  012737  000001  002320        MOV     #1,PR.BIT       ;ELSE INSERT A 1
6654
6655  031204  004737  033414        1$:     JSR     PC,RDBIT        ;GO READ BIT
6656  031210  016237  000026  002204        MOV     RKMR1(R2),T.MR1 ;GET MR1
6657  031216  023737  002204  002244        CMP     T.MR1,E.MR1     ;CHECK IF CORRECT
6658  031224  001402                        BEQ     43$             ;YES - SKIP
6659  031226  000137  032060                JMP     9$
6660
6661  031232  005737  002326        43$:    TST     BITCNT          ;TEST IF FIRST BIT OF DATA
6662  031236  001413                        BEQ     44$             ;YES - SKIP ECC GEN
6663  031240  004737  032260                JSR     PC,ECCGEN       ;GO GENERATE ECC
6664  031244  016237  000032  002214        MOV     RKECPT(R2),T.ECPT  ;TEST IF ECC CORRECT
6665  031252  023737  002214  002254        CMP     T.ECPT,E.ECPT
6666  031260  001402                        BEQ     44$
6667  031262  000137  032074                JMP     10$
6668
6669  031266  005237  002326        44$:    INC     BITCNT          ;BUMP BIT COUNTER
6670  031272  005701                        TST     R1              ;TEST IF THIS WORD IS DONE
6671  031274  100402                        BMI     2$              ;YES - SKIP
6672  031276  006301                        ASL     R1              ;ELSE SHIFT BIT POINTER
6673  031300  000727                        BR      4$
6674
6675  031302  032737  010000  002260  2$:   BIT     #CFMT,L.CS1     ;TEST IF 18 BIT MODE
6676  031310  001412                        BEQ     30$             ;NO - SKIP
```

```
 6677  031312  005737  001214            TST     $TMP5           ;TEST IF PROCESSING BIT 16 OR 17 DONE
 6678  031316  001007                    BNE     30$             ;YES - SKIP
 6679  031320  012701  040000            MOV     #BIT14,R1       ;SET POINTER FOR 2 MORE BITS
 6680  031324  013705  002354            MOV     HIBITS,R5       ;INSERT HI ORDER BITS
 6681  031330  005137  001214            COM     $TMP5           ;SET 18 BIT SWITCH
 6682  031334  000711                    BR      4$
 6683  031336  005037  001214    30$:    CLR     $TMP5           ;CLEAR 18 BIT SWITCH
 6684
 6685  031342  005300            31$:    DEC     R0              ;DECREMENT WORD COUNT
 6686  031344  001302                    BNE     3$              ;IF NOT ZERO - LOOP
 6687
 6688  031346  023737  002326  001226    CMP     BITCNT,$TMP12   ;ELSE TEST IF FULL SECTOR WAS READ
 6689  031354  001402                    BEQ     33$             ;YES - SKIP
 6690  031356  000137  032032            JMP     8$              ;ELSE JUMP TO EXIT - DO NOT CHECK
 6691                                                            ;    ECC WORDS
 6692
 6693  031362  013737  002320  002322  33$:    MOV     PR.BIT,M1.BIT   ;MOVE LAST DATA BIT FOR ECC GEN
 6694  031370  004737  032260            JSR     PC,ECCGEN       ;GENERATE ECC
 6695  031374  005037  001214            CLR     $TMP5           ;CLEAR FOR PASS COUNTING
 6696  031400  005737  002344            TST     ECCSRC          ;TEST WHERE ECC WORDS ARE
 6697  031404  001006                    BNE     36$             ;THEY ARE IN BUFFER - SKIP
 6698  031406  013737  002340  052444    MOV     ECCHI,OBUFF+1000        ;ELSE MOVE THEM INTO BUFFER
 6699  031414  013737  002342  052446    MOV     ECCLO,OBUFF+1002
 6700
 6701  031422  012305            36$:    MOV     (R3)+,R5        ;GET ECC FROM BUFFER
 6702  031424  012701  000001            MOV     #BIT0,R1        ;SET BIT POINTER
 6703
 6704  031430  013737  002320  002322  5$:     MOV     PR.BIT,M1.BIT   ;SHIFT PRESENT TO PREVIOUS
 6705  031436  005037  002320            CLR     PR.BIT          ;AND SET UP PRESENT
 6706  031442  030105                    BIT     R1,R5
 6707  031444  001403                    BEQ     34$
 6708  031446  012737  000001  002320    MOV     #1,PR.BIT
 6709  031454  004737  033414    34$:    JSR     PC,RDBIT        ;GO READ BIT
 6710  031460  016237  000026  002204    MOV     RKMR1(R2),T.MR1 ;GET MR1
 6711  031466  023737  002204  002244    CMP     T.MR1,E.MR1     ;CHECK IF CORRECT
 6712  031474  001402                    BEQ     38$             ;YES - SKIP
 6713  031476  000137  032110            JMP     11$             ;ELSE JUMP TO REPORT EXIT
 6714  031502  023737  002326  001226  38$:    CMP     BITCNT,$TMP12   ;TEST IF FIRST ECC BIT
 6715  031510  001402                    BEQ     49$             ;YES - SKIP ECC GEN (ALREADY DONE)
 6716  031512  004737  032260            JSR     PC,ECCGEN       ;GENERATE ECC FOR BIT
 6717  031516  016237  000032  002214  49$:    MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
 6718  031524  023737  002214  002254    CMP     T.ECPT,E.ECPT   ;TEST IF ECC PAT CORRECT
 6719  031532  001402                    BEQ     45$             ;YES SKIP
 6720  031534  000137  032074            JMP     10$             ;ELSE GO REPORT ERROR
 6721
 6722  031540  005237  002326    45$:    INC     BITCNT          ;BUMP BIT COUNT
 6723  031544  023737  002326  001230    CMP     BITCNT,$TMP13   ;TEST IF 1ST BIT OF ECC
 6724  031552  001003                    BNE     46$             ;NO - SKIP
 6725  031554  042737  020000  002244    BIC     #ECCW,E.MR1     ;ELSE CLR ECCW FOR ECC READ
 6726  031562  023737  002326  001232  46$:    CMP     BITCNT,$TMP14   ;TEST IF 1ST BIT OF POSTAMBLE
 6727  031570  001027                    BNE     47$             ;NO - SKIP
 6728  031572  052737  020000  002244    BIS     #ECCW,E.MR1     ;ELSE SET ECCW FOR ECC DONE
 6729  031600  042737  100000  002244    BIC     #RDGATE,E.MR1   ;CLEAR READ GATE
 6730
 6731  031606  016237  000032  002214    MOV     RKECPT(R2),T.ECPT ;STORE ACTUAL PATTERN
 6732  031614  016237  000030  002212    MOV     RKECPS(R2),T.ECPS ;STORE ACTUAL POSITION
```

```
6733  031622  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;GET EXPECTED POSITION FOR REPORT
6734  031630  023737  002214  002254          CMP     T.ECPT,E.ECPT   ;CHECK IF CORRECT
6735  031636  001163                          BNE     20$             ;NO SKIP
6736  031640  023737  002212  002252          CMP     T.ECPS,E.ECPS   ;CHECK IF POSITION CORRECT
6737  031646  001157                          BNE     20$             ;NO - SKIP
6738
6739  031650  023737  002326  001232  47$:    CMP     BITCNT,$TMP14   ;TEST IF ECC DONE
6740  031656  002436                          BLT     48$             ;NO - SKIP
6741  031660  005737  002346                  TST     ECPATX          ;TEST IF EXPECTED ECC IS 0
6742                                                                  ;(NO ERROR)
6743  031664  001433                          BEQ     48$             ;YES - SKIP
6744  031666  023737  001232  002326          CMP     $TMP14,BITCNT   ;TEST IF FIRST POSTAMBLE BIT
6745  031674  001402                          BEQ     52$             ;YES - SKIP
6746  031676  005237  002350                  INC     ECPOSX          ;ELSE BUMP POS COUNT EXPECTED
6747  031702  016237  000032  002214  52$:    MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
6748  031710  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
6749  031716  013737  002350  002252          MOV     ECPOSX,E.ECPS   ;GET EXPECTED POSITION
6750  031724  023737  002214  002254          CMP     T.ECPT,E.ECPT   ;TEST IF PATTERN CORRECT
6751  031732  001402                          BEQ     51$             ;YES - SKIP
6752  031734  000137  032124                  JMP     22$             ;ELSE GO REPORT
6753  031740  023737  002212  002252  51$:    CMP     T.ECPS,E.ECPS   ;CHECK IF POSITION CORRECT
6754  031746  001402                          BEQ     48$             ;YES - SKIP
6755  031750  000137  032140                  JMP     23$             ;ELSE GO REPORT
6756
6757  031754  005701                  48$:    TST     R1              ;TEST IF WORD DONE
6758  031756  100403                          BMI     6$              ;YES SKIP
6759  031760  006301                          ASL     R1              ;ELSE SHIFT BIT POINTER
6760  031762  000137  031430                  JMP     5$              ;AND LOOP
6761
6762  031766  005737  001214          6$:     TST     $TMP5           ;TEST PASS COUNT. IF NOT 0
6763                                                                  ;(NOT THE FIRST ECC WORD JUST WRITTE)
6764  031772  001004                          BNE     7$              ; - SKIP
6765
6766  031774  005237  001214                  INC     $TMP5           ;BUMP PASS COUNT
6767  032000  000137  031422                  JMP     36$
6768
6769  032004  023727  001214  000001  7$:     CMP     $TMP5,#1        ;TEST PASS COUNT. IF NOT 1 (NOT THE
6770  032012  001007                          BNE     8$              ;2ND ECC WORD JUST READ) - SKIP
6771  032014  005005                          CLR     R5              ;CLEAR WORD (POSTAMBLE)
6772  032016  012701  000001                  MOV     #BIT0,R1        ;SET BIT POINTER
6773  032022  005237  001214                  INC     $TMP5           ;BUMP PASS COUNT
6774  032026  000137  031430                  JMP     5$
6775
6776  032032  004437  034154          8$:     JSR     R4,GETREG       ;GET RK611 REGS
6777  032036  013737  002260  002220          MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
6778  032044  023737  002220  002160          CMP     E.CS1,T.CS1     ;CHECK IF CORRECT
6779  032052  001060                          BNE     21$             ;NO - SKIP
6780  032054  005724                          TST     (R4)+           ;ELSE BUMP RETURN POINTER TO NO ERR
6781  032056  000446                          BR      13$
6782
6783  032060  013704  001216          9$:     MOV     $TMP6,R4        ;SET RETURN TOR MR1 ERR.
6784  032064  012737  031232  001236          MOV     #43$,$ESCAPE    ;SET RETURN TO CONTINUE TEST
6785  032072  000427                          BR      12$
6786
6787  032074  013704  001220          10$:    MOV     $TMP7,R4        ;SET RETURN FOR ECC ERROR
6788  032100  012737  031266  001236          MOV     #44$,$ESCAPE    ;SET RETURN TO CONTINUE TEST
```

```
6789  032106  000421                          BR      12$
6790
6791  032110  013704  001216          11$:    MOV     $TMP6,R4        ;SET RETURN FOR  MR1 ERROR
6792  032114  012737  031540  001236          MOV     #45$,$ESCAPE    ;SET RETURN TO CONTINUE TEST
6793  032122  000413                          BR      12$
6794
6795  032124  013704  001222          22$:    MOV     $TMP10,R4       ;GET RETURN FOR PATTERN ERROR
6796  032130  012737  031740  001236          MOV     #51$,$ESCAPE    ;SET RETURN TO CONTINUE TEST
6797  032136  000405                          BR      12$
6798
6799  032140  013704  001224          23$:    MOV     $TMP11,R4       ;RET RETURN FOR POSITION ERROR
6800  032144  012737  031754  001236          MOV     #48$,$ESCAPE    ;SET RETURN TO CONTINUE TEST
6801
6802
6803  032152  013746  001224          12$:    MOV     $TMP11,-(SP)    ;PREP STACK FOR RETURN
6804  032156  032777  001000  146754          BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
6805  032164  001407                          BEQ     14$             ;NO SKIP
6806
6807  032166  005726                          TST     (SP)+           ;REMOVE LAST ENTRY FROM STACK
6808  032170  005037  001236                  CLR     $ESCAPE         ;NO LOOP - SLEAR ESCAPE
6809
6810  032174                          13$:
6811  032174  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
6812  032176  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
6813  032200  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
6814  032202  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
6815  032204  000204                  14$:    RTS     R4
6816
6817  032206  013704  001212          20$:    MOV     $TMP4,R4        ;GET RETURN FOR ECC ERROR
6818  032212  000402                          BR      15$
6819  032214  013704  001210          21$:    MOV     $TMP3,R4        ;GET CS1 ERROR RETURN
6820  032220  005037  001236          15$:    CLR     $ESCAPE         ;TEST IF LOOP ON ERROR
6821  032224  032777  001000  146706          BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
6822  032232  001360                          BNE     13$             ;YES - SKIP
6823
6824  032234  013700  001254                  MOV     $TESTN,R0       ;SET REGISTERS TO LOOP ON ERROR OR
6825  032240  006300                          ASL     R0              ;ESCAPE TO NEXT TEST.
6826  032242  016037  035270  001236          MOV     $SW08TB(R0),$ESCAPE     ;SET UP TO ESCAPE TO NEXT TEST
6827
6828  032250  162737  000002  001236          SUB     #2,$ESCAPE      ;ADJUST RETURN
6829  032256  000746                          BR      13$             ;GO TO EXIT
6830
6831                                  .SBTTL  GENERATE ECC WORD
6832                                  ;*      EACH TIME THIS ROUTINE IS CALLED THE ECC IS CALCULATED FOR THE
6833                                  ;*      NEXT BIT TO BE READ OR WRITTEN AND THE BITS PREVIOUSLY
6834                                  ;*      READ OR WRITTEN (STORED IN ECCHI AND ECCLO). THE RESULTS OF
6835                                  ;*      THE CALCULATION ARE PLACED IN ECCHI AND ECCLO. THE LOW ORDER
6836                                  ;*      11 BITS OF ECCHI ARE PLACED IN E.ECPT FOR EASY COMPARISON
6837                                  ;*      TO RKECPT.
6838                                  ;*
6839                                  ;*      CALL:   JSR     PC,ECCGEN
6840                                  ;*      RETURN: RTS     R4
6841  032260  010046                  ECCGEN: MOV     R0,-(SP)        ;STORE R0
6842  032262  012700  002316                  MOV     #P1.BIT,R0      ;GET ADDRESS OF NEXT BIT
6843  032266  032737  000002  002260          BIT     #BIT1,L.CS1     ;CHECK IF WRITING
6844  032274  001002                          BNE     12$             ;YES - SKIP
```

```
6845  032276  012700  002322                MOV     #M1.BIT,R0      ;ELSE CHANGE TO PRESENT BIT
6846
6847  032302  005710                  12$:   TST     (R0)            ;CHECK IF NEXT BIT 0
6848  032304  001410                         BEQ     3$              ;YES - SKIP
6849  032306  032737  000001  002340         BIT     #1,ECCHI        ;NO, CHECK IF BIT 31 = 1
6850  032314  001410                         BEQ     5$              ;NO, SET ECC XORED BIT
6851  032316  005037  002336          1$:    CLR     ECCXOR          ;CLEAR ECC XORED BIT
6852  032322  000241                         CLC                     ;CLEAR CARRY FLOP
6853  032324  000410                         BR      7$              ;SHIFT IN ECC BIT
6854
6855  032326  032737  000001  002340  3$:    BIT     #1,ECCHI        ;CHECK IF BIT 31 = 1
6856  032334  001770                         BEQ     1$              ;NO, CLEAR ECC XORED BIT
6857  032336  012737  000001  002336  5$:    MOV     #1,ECCXOR       ;SET ECC XOR
6858  032344  000261                         SEC                     ;SET CARRY FLOP
6859  032346  006037  002342          7$:    ROR     ECCLO
6860  032352  006037  002340                 ROR     ECCHI
6861  032356  005737  002336                 TST     ECCXOR          ;CHECK IF XOR NEEDED
6862  032362  001422                         BEQ     10$             ;NO, RETURN
6863  032364  012746  020020                 MOV     #BIT13!BIT4,-(SP) ;DO XOR OF ECC BITS
6864  032370  043716  002342                 BIC     ECCLO,(SP)      ;  0, 2, 11, 21, 23
6865  032374  042737  020020  002342         BIC     #BIT13!BIT4,ECCLO
6866  032402  052637  002342                 BIS     (SP)+,ECCLO
6867  032406  012746  002400                 MOV     #BIT10!BIT8,-(SP)
6868  032412  043716  002340                 BIC     ECCHI,(SP)
6869  032416  042737  002400  002340         BIC     #BIT10!BIT8,ECCHI
6870  032424  052637  002340                 BIS     (SP)+,ECCHI
6871  032430  013737  002340  002254  10$:    MOV     ECCHI,E.ECPT    ;STORE ECC PATTERN
6872  032436  042737  174000  002254         BIC     #174000,E.ECPT  ;MASK UNSEEN BITS
6873  032444  012600                         MOV     (SP)+,R0        ;RESTORE R0
6874  032446  000207                         RTS     PC              ;RETURN
6875
6876                                  .SBTTL  SIMULATE ONE BIT OF WRITE DATA IN MAINTANENCE MODE
6877                                  ;*      ONE BIT OF DATA IS WRITE SIMULATED. THE CONTENTS OF
6878                                  ;*      RKMR1 IS COMPUTED (PRECOMP ADVANCE AND DELAY, MAINTENANCE
6879                                  ;*      ENCODED WRITE DATA) FOR THE PROPER TRANSITION OF THE
6880                                  ;*      MAINTENANCE CLOCK AND CHECKED AT EACH TRANSITION. IF MR1
6881                                  ;*      IS INCORRECT AT ANY TRANSITION, AN ERROR IS SET UP TO
6882                                  ;*      CONTAIN A MESSAGE THAT IDENTIFIES THE TRANSITION WHEN THE
6883                                  ;*      ERROR OCCURRED.
6884                                  ;*
6885                                  ;*      CALL:   JSR     PC,WRTBIT
6886                                  ;*      RETURN: RTS     PC+2            FOR NO ERROR RETURN
6887                                  ;*              RTS     PC              FOR ERROR IN MR1
6888  032450  052737  002400  002244  WRTBIT: BIS     #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT. REG. 1
6889  032456  012762  000440  000026         MOV     #DMD!MCLK,RKMR1(R2) ;PROVIED 1ST UPWARD TRANSITION
6890  032464  016237  000026  002204         MOV     RKMR1(R2),T.MR1 ;STORE MAINT. REG. 1
6891  032472  023737  002244  002204         CMP     E.MR1,T.MR1     ;CHECK IF MAINT REG 1 CORRECT
6892  032500  001416                         BEQ     3$              ;YES, PROVIDE DOWNWARD TRANSISTION
6893  032502  012737  032522  001236         MOV     #1$,$ESCAPE     ;LOAD ESCAPE FOR LOOP ON ERROR
6894  032510  012737  045452  001512         MOV     #EMW1,EMW+2     ;LOAD ERROR MESSAGE
6895  032516  011646                         MOV     (SP),-(SP)      ;SAVE RETURN
6896  032520  000207                         RTS     PC              ;MR1 INCORRECT ON UPWARD TRANSITION
6897
6898  032522  032777  001000  146410  1$:    BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6899  032530  001402                         BEQ     3$              ;NO, CONTINUE
6900  032532  000137  033404                 JMP     63$             ;YES, LOOP ON ERROR
```

```
6901
6902  032536  042737  014400  002244  3$:     BIC     #MCLK!PCA!PCD,E.MR1 ;INITIALIZE MAINTENANCE REG. 1
6903  032544  052737  042000  002244          BIS     #MEWD!WRTGAT,E.MR1
6904  032552  005737  002320                  TST     PR.BIT          ;CHECK IF ONE
6905  032556  001152                          BNE     20$             ;YES, SIMULATE ONE
6906  032560  005737  002322                  TST     M1.BIT          ;CHECK IF PREVIOUS ONE
6907  032564  001023                          BNE     10$             ;YES, NO TRANSISTION
6908  032566  042737  002000  002244          BIC     #MEWD,E.MR1     ;INDICATE TRANSISTION
6909  032574  005737  002316                  TST     P1.BIT          ;CHECK IF NEXT BIT = 1
6910  032600  001007                          BNE     5$              ;YES, CHECK FOR PRECOMP ADVANCE
6911  032602  005737  002324                  TST     M2.BIT          ;CHECK FOR PRECOMP. ADVANCE
6912  032606  001412                          BEQ     10$             ;NO, CLOCK IN ZERO
6913  032610  052737  010000  002244          BIS     #PCD,E.MR1      ;SET PRECOMP. DELATY
6914  032616  000406                          BR      10$             ;CLOCK IN ZERO
6915
6916  032620  005737  002324          5$:     TST     M2.BIT          ;CHECK FOR PRECOMP. ADVANCE
6917  032624  001003                          BNE     10$             ;CLOCK IN ZERO
6918  032626  052737  004000  002244          BIS     #PCA,E.MR1      ;SET PRECOMP. ADVANCE
6919  032634  012762  000040  000026  10$:     MOV     #DMD,RKMR1(R2)  ;CLOCK IN DATA BIT
6920  032642  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MR1
6921  032650  023737  002204  002244          CMP     T.MR1,E.MR1     ;CHECK IF MR1 CORRECT
6922  032656  001416                          BEQ     12$             ;YES, CONTINUE
6923  032660  012737  032700  001236          MOV     #11$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
6924  032666  012737  045540  001512          MOV     #EMW2,EMW+2     ;LOAD ERROR MESSAGE
6925  032674  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
6926  032676  000207                          RTS     PC              ;MR1 INCORRECT
6927
6928  032700  032777  001000  146232  11$:     BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6929  032706  001402                          BEQ     12$             ;NO, CONTINUE
6930  032710  000137  033404                  JMP     63$             ;YES, LOOP ON ERROR
6931
6932  032714  052737  002400  002244  12$:     BIS     #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT REG 1
6933  032722  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSISITION
6934  032730  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MAINT REG. 1
6935  032736  023737  002244  002204          CMP     E.MR1,T.MR1     ;CHECK IF MAINT REG. 1 CORRECT
6936  032744  001416                          BEQ     15$             ;YES, CONTINUE
6937  032746  012737  032766  001236          MOV     #13$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
6938  032754  012737  045630  001512          MOV     #EMW3,EMW+2     ;LOAD ERROR MESSAGE
6939  032762  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
6940  032764  000207                          RTS     PC              ;MR1 INCORRECT
6941
6942  032766  032777  001000  146144  13$:     BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6943  032774  001402                          BEQ     15$             ;NO, CONTINUE
6944  032776  000137  033404                  JMP     63$             ;YES, LOOP ON ERROR
6945
6946  033002  052737  002000  002244  15$:     BIS     #MEWD,E.MR1     ;RESET TRANSITION INDICATION
6947  033010  042737  000400  002244          BIC     #MCLK,E.MR1
6948  033016  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;SUPPLY LAST PART OF DATA
6949  033024  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MR1
6950  033032  023737  002204  002244          CMP     T.MR1,E.MR1     ;CHECK IF MR1 CORRECT
6951  033040  001414                          BEQ     18$             ;YES, RETURN
6952  033042  012737  033062  001236          MOV     #17$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
6953  033050  012737  045716  001512          MOV     #EMW4,EMW+2     ;LOAD ERROR MESSAGE
6954  033056  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
6955  033060  000207                          RTS     PC              ;MR1 INCORRECT
6956
```

```
6957  033062  032777  001000  146050  17$:    BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6958  033070  001145                          BNE     63$             ;YES, LOOP ON ERROR
6959  033072  005037  001236          18$:    CLR     $ESCAPE         ;CLEAR ESCAPE
6960  033076  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN
6961  033102  000207                          RTS     PC              ;RETURN
6962
6963  033104  005737  002316          20$:    TST     P1.BIT          ;CHECK IN NEXT BIT A ONE
6964  033110  001007                          BNE     30$             ;YES, CHECK IF PRECOMP DELAY
6965  033112  005737  002322                  TST     M1.BIT          ;CHECK FOR PRECOMP ADVANCE
6966  033116  001415                          BEQ     40$             ;NO, CHECK FO MR1
6967  033120  052737  004000  002244          BIS     #PCA,E.MR1      ;SET PRECOMP. ADVZNCE
6968  033126  000411                          BR      40$             ;CHECK MR1
6969
6970  033130  042737  000400  002244  30$:    BIC     #MCLK,E.MR1     ;RESET MAINT CLOCK IN EXPECTED MR1
6971  033136  005737  002322                  TST     M1.BIT          ;CHECK FOR PRECOMP DELAY
6972  033142  001003                          BNE     40$             ;NO, CHECK MR1
6973  033144  052737  010000  002244          BIS     #PCD,E.MR1      ;SET SET PRECOMP DELAY
6974  033152  012762  000040  000026  40$:    MOV     #DMD,RKMR1(R2)  ;CLOCK IN DATA BIT
6975  033160  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MR1
6976  033166  023737  002204  002244          CMP     T.MR1,E.MR1     ;CHECK MR1 CORRECT
6977  033174  001414                          BEQ     42$             ;YES, CLOCK IN REST OF BIT
6978  033176  012737  033216  001236          MOV     #41$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
6979  033204  012737  045540  001512          MOV     #EMW2,EMW+2     ;LOAD ERROR MESSAGE
6980  033212  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
6981  033214  000207                          RTS     PC              ;MR1 INCORRECT
6982
6983  033216  032777  001000  145714  41$:    BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6984  033224  001067                          BNE     63$             ;YES, LOOP ON ERROR
6985  033226  052737  000400  002244  42$:    BIS     #MCLK,E.MR1     ;CHREATE EXPECTED MAINT. REG. 1
6986  033234  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2) ;PROVIDE 2ND UPWARD TRANSITION
6987  033242  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MAINT REG 1
6988  033250  023737  002244  002204          CMP     E.MR1,T.MR1     ;CHECK IF MAINT REG 1 CORRECT
6989  033256  001414                          BEQ     45$             ;YES, CONTINUE
6990  033260  012737  033300  001236          MOV     #43$,$ESCAPE    ;LOAD ESCAPE
6991  033266  012737  045630  001512          MOV     #EMW3,EMW+2     ;LOAD ERROR MESSAGE
6992  033274  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
6993  033276  000207                          RTS     PC              ;MR1 INCORRECT
6994
6995  033300  032777  001000  145632  43$:    BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
6996  033306  001036                          BNE     63$             ;YES, LOOP ON ERROR
6997  033310  042737  002400  002244  45$:    BIC     #MEWD!MCLK,E.MR1 ;SET TRANSISTION
6998  033316  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;CLOCK TRANSISTION
6999  033324  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;STORE MR1
7000  033332  023737  002204  002244          CMP     T.MR1,E.MR1     ;CHECK IF MR1 CORRECT
7001  033340  001414                          BEQ     50$             ;YES, RETURN
7002  033342  012737  033362  001236          MOV     #47$,$ESCAPE    ;LOAD ESCAPE FOR LOOP ON ERROR
7003  033350  012737  045716  001512          MOV     #EMW4,EMW+2     ;LOAD ERROR MESSAGE
7004  033356  011646                          MOV     (SP),-(SP)      ;SAVE RETURN
7005  033360  000207                          RTS     PC              ;MR1 INCORRECT
7006
7007  033362  032777  001000  145550  47$:    BIT     #SW9,@SWR       ;CHECK IF LCOP ON ERROR
7008  033370  001005                          BNE     63$             ;YES, LOOP ON ERROR
7009  033372  005037  001236          50$:    CLR     $ESCAPE         ;CLEAR ESCAPE
7010  033376  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN
7011  033402  000207                          RTS     PC              ;RETURN
7012
```

```
7013  033404  012706  001100          63$:    MOV     #STACK,SP       ;FORCE STACK
7014  033410  000177  145474                  JMP     @$LPERR         ;LOOP ON ERROR
7015
7016                                  .SBTTL  SIMULATE ONE BIT OR READ DATA IN MAINTENANCE MODE
7017                                  ;*      ONE BIT OF DATA IS READ SIMULATED. NO CHECKING IS
7018                                  ;*      DONE BY THIS ROUTINE.
7019                                  ;*
7020                                  ;*      CALL:   JSR     PC,RDBIT
7021                                  ;*      RETURN: RTS     PC
7022  033414  005737  002320          RDBIT:  TST     PR.BIT          ;CHECK IF ONE
7023  033420  001024                          BNE     10$             ;YES, SIMULATE ONE
7024  033422  005737  002322                  TST     M1.BIT          ;CHECK IF PREVIOUS ONE
7025  033426  001404                          BEQ     4$              ;NO, INSERT TRANSITION
7026  033430  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2)  ;YES, DO NOT INSERT TRANSITION
7027  033436  000403                          BR      5$              ;CLOCK IN ZERO
7028
7029  033440  012762  001440  000026  4$:     MOV     #DMD!MCLK!MERD,RKMR1(R2)  ;INSERT TRANSITION
7030  033446  012762  000040  000026  5$:     MOV     #DMD,RKMR1(R2)  ;CLOCK IN ZERO
7031  033454  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2)
7032  033462  012762  000040  000026          MOV     #DMD,RKMR1(R2)
7033  033470  000207                          RTS     PC              ;RETURN
7034
7035  033472  012762  000440  000026  10$:    MOV     #DMD!MCLK,RKMR1(R2)  ;CLOCK IN ONE
7036  033500  012762  000040  000026          MOV     #DMD,RKMR1(R2)
7037  033506  012762  001440  000026          MOV     #DMD!MCLK!MERD,RKMR1(R2)
7038  033514  012762  000040  000026          MOV     #DMD,RKMR1(R2)
7039  033522  000207                          RTS     PC              ;RETURN
7040
7041                                  .SBTTL  MEMORY CHECK ENABLE TRAP
7042                                  ;*      IF THE PROCESSOR IN USE HAS MEMORY PARITY OPTION,
7043                                  ;*      IT WILL BE ENABLED. THIS ROUTINE WILL PROCESS TRAPS CAUSED
7044                                  ;*      BY MEMORY PARITY ERRORS.
7045  033524  012737  033540  001236  MEMERR: MOV     #10$,$ESCAPE    ;LOAD ESCAPE
7046  033532  011637  002310                  MOV     (SP),TRAPPC     ;STORE PC
7047  033536  104001                          ERROR   1               ;REPORT MEM PARITY ERROR
7048  033540  005037  001236          10$:    CLR     $ESCAPE         ;CLEAR ESCAPE
7049  033544  032777  001000  145366          BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
7050  033552  001001                          BNE     15$             ;YES, FORCE STACK AND TRY AGAIN
7051  033554  000002                          RTI                     ;NO, RETURN
7052
7053  033556  012706  001100          15$:    MOV     #STACK,SP       ;INITIALIZE STACK
7054  033562  000177  145322                  JMP     @$LPERR         ;LOOP ON ERROR
7055
7056                                  .SBTTL  CLEAR INPUT BUFFER
7057  033566  012700  050440          CLRIBF: MOV     #IBUFF,R0       ;GET BUFFER POINTER
7058  033572  010146                          MOV     R1,-(SP)        ;STORE R1
7059  033574  012701  000400                  MOV     #400,R1 ;SET COUNT FOR CLEAR
7060  033600  005020                  1$:     CLR     (R0)+           ;CLEAR BUFFER WORD
7061  033602  005301                          DEC     R1              ;DEC COUNT
7062  033604  001375                          BNE     1$              ;LOOP UNTIL COUNT IS ZERO
7063  033606  012601                          MOV     (SP)+,R1        ;RESTORE R1
7064  033610  000204                          RTS     R4              ;RETURN
7065                                  .SBTTL  BUILD DATA BUFFER
7066                                  ;*      THE PATTERN SPECIFIED IN THE CALL IS LOADED INTO THE DATA
7067                                  ;*      BUFFER. THE ENTIRE BUFFER IS ALWAYS LOADED (400(8) WORDS).
7068  033612                          BLDDAT:
```

```
7069  033612  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7070  033614  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7071  033616  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7072  033620  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7073  033622  012700  051444                MOV     #OBUFF,R0       ;LOAD R0 TO POINT TO BUFFER
7074  033626  012701  000400                MOV     #400,R1         ;SET DATA LENGTH
7075  033632  021427  000001                CMP     (R4),#1         ;TEST IF PATTERN 1(ALL ZERO)
7076  033636  001004                        BNE     2$              ;NO - SKIP
7077  033640  005020                1$:     CLR     (R0)+           ;ELSE CLEAR OBUFF
7078  033642  005301                        DEC     R1
7079  033644  001375                        BNE     1$
7080  033646  000461                        BR      13$
7081  033650  021427  000007          2$:   CMP     (R4),#7         ;TEST IF PAT 7 (ALL ONES)
7082  033654  001006                        BNE     4$              ;NO SKIP
7083  033656  012703  177777                MOV     #177777,R3      ;ELSE SET BUFFER TO ALL ONES
7084  033662  010320                3$:     MOV     R3,(R0)+
7085  033664  005301                        DEC     R1
7086  033666  001375                        BNE     3$
7087  033670  000450                        BR      13$
7088  033672  021427  000006          4$:   CMP     (R4),#6         ;TEST IF PAT 6 (COMPOSIT ROTATING)
7089  033676  001003                        BNE     5$              ;NO - SKIP
7090  033700  012705  050400                MOV     #PAT6,R5        ;ELSE GET ADDRESS OF PATTERN 6
7091  033704  000427                        BR      10$             ;GO LOAD BUFFER
7092  033706  021427  000005          5$:   CMP     (R4),#5         ;TEST IF PATTERN 5 (ALT 1 & 0)
7093  033712  001006                        BNE     7$              ;NO SKIP
7094  033714  012703  125252                MOV     #125252,R3      ;ELSE LOAD OBUFF WITH PAT 5
7095  033720  010320                6$:     MOV     R3,(R0)+
7096  033722  005301                        DEC     R1
7097  033724  001375                        BNE     6$
7098  033726  000431                        BR      13$
7099  033730  021427  000004          7$:   CMP     (R4),#4         ;TEST IF PATTERN 4 (HI-LO FREQ MIX)
7100  033734  001003                        BNE     8$              ;NO - SKIP
7101  033736  012705  050340                MOV     #PAT4,R5        ;LOAD POINTER WITH ADD OF PAT 4
7102  033742  000410                        BR      10$
7103  033744  021427  000003          8$:   CMP     (R4),#3         ;TEST IF PAT 3 (MAX PRECOMP PHASE MIX)
7104  033750  001003                        BNE     9$              ;NO - SKIP
7105  033752  012705  050300                MOV     #PAT3,R5        ;LOAD POINTER WITH ADD OF PAT 3
7106  033756  000402                        BR      10$             ;GO LOAD BUFFER
7107  033760  012705  050240          9$:   MOV     #PAT2,R5        ;GET ADDRESS OF PAT 2
7108  033764  012703  000020          10$:  MOV     #16.,R3         ;SET PATTERN LENGTH COUNT
7109  033770  012520                11$:    MOV     (R5)+,(R0)+     ;MOV PATTERN INTO OBUFF
7110  033772  005301                        DEC     R1              ;DEC COUNTERS
7111  033774  005303                        DEC     R3
7112  033776  001374                        BNE     11$             ;LOOP UNTIL PATTERN IS MOVED
7113  034000  012705  051444                MOV     #OBUFF,R5       ;SET R5 TO START OF BUFFER
7114  034004  012520                12$:    MOV     (R5)+,(R0)+     ;REPEAT PATTERN THROUGH BUFFER
7115  034006  005301                        DEC     R1
7116  034010  001375                        BNE     12$
7117  034012  005724                13$:    TST     (R4)+           ;BUMP PAST PARAMETER
7118  034014  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
7119  034016  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
7120  034020  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
7121  034022  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
7122  034024  000204                        RTS     R4
7123
7124                                  .SBTTL  LOAD "L" REGISTERS
```

```
                                        ;*      THE "L" REGISTERS ARE LOADED FORM THE PARAMETERS FOLLOWING
                                        ;*      THE SUBROUTINE CALL. DIAGNOSTIC MODE IS SET IN L.MR1
                                        ;*      AND L.CS1 IS LOADED WITH THE COMMAND.
                                        ;*
                                        ;*      CALL:
                                        ;*      JSR     R4,LOADRK
                                        ;*      .WORD           CYLINDER
                                        ;*      .BYTE           ;SECTOR
                                        ;*      .BYTE           ;TRACK
                                        ;*      .WORD           ;BUFFER ADDRESS
                                        ;*      .WORD           ;WORD COUNT
                                        ;*      .WORD           ;COMMAND,FORMAT,DRIVE TYPE,&UPPER BUS ADDRESS BITS
                                        ;*
                                        ;*      RETURN:
                                        ;*      RTS     R4

034026  012437  002274          LOADRK: MOV     (R4)+,L.DCYL            ;LOAD CYLINDER
034032  012437  002266                  MOV     (R4)+,L.DA              ;       TRACK AND SECTOR
034036  012437  002264                  MOV     (R4)+,L.BA              ;       BUS ADDRESS
034042  012437  002262                  MOV     (R4)+,L.WC              ;       WORD COUNT
034046  012737  000040  002276          MOV     #DMD,L.MR1              ;SET DIAGNOSTIC MODE
034054  005037  002272                  CLR     L.ASOF                  ;CLEAR OFFSET
034060  005037  002270                  CLR     L.CS2                   ;CLEAR CS2
034064  012437  002260                  MOV     (R4)+,L.CS1             ;LOAD CS1
034070  000204                          RTS     R4                      ;RETURN

                                .SBTTL  START THE OPERATION
                                ;*      THE INFORMATION IN THE "L" REGISTERS ARE LOADED INTO THE
                                ;*      RK611 REGISTERS IN A STARAIGHT TRANSFER.

034072  013762  002262  000002  OPSTRT: MOV     L.WC,RKWC(R2)   ;MOVE THE "L"REGISTERS INTO
034100  013762  002264  000004          MOV     L.BA,RKBA(R2)   ;CORRESPONDING RK611 REGISTERS.
034106  013762  002274  000020          MOV     L.DCYL,RKDCYL(R2)
034114  013762  002266  000006          MOV     L.DA,RKDA(R2)
034122  013762  002270  000010          MOV     L.CS2,RKCS2(R2)
034130  013762  002276  000026          MOV     L.MR1,RKMR1(R2)
034136  013762  002272  000016          MOV     L.ASOF,RKASOF(R2)
034144  013762  002260  000000          MOV     L.CS1,RKCS1(R2)
034152  000204                          RTS     R4

                                .SBTTL  GET THE RK611 REGISTERS
                                ;*      ALL THE RK611 REGISTERS EXCEPT THE DATA BUFFER ARE
                                ;*      STORED IN THE "T" REGISTERS.

034154                          GETREG:
034154  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
034156  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
034160  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
034162  010200                          MOV     R2,R0           ;GET RK BASE
034164  012701  002160                  MOV     #T.CS1,R1       ;GET START OF REGS
034170  012703  000011                  MOV     #11,R3          ;SET COUNT
034174  012021                  1$:     MOV     (R0)+,(R1)+     ;GET CS1 THRU DCYL
034176  005303                          DEC     R3
034200  001375                          BNE     1$              ;LOOP UNTIL DONE

034202  062700  000004                  ADD     #4,R0           ;SKIP OVER DB AND SPARE
```

```
7181  034206  005021                        CLR     (R1)+           ;CLEAR T.DB
7182  034210  012021                        MOV     (R0)+,(R1)+     ;STORE MR1
7183  034212  012037  002212                MOV     (R0)+,T.ECPS    ;       ECC POSITION
7184  034216  012037  002214                MOV     (R0)+,T.ECPT    ;       ECC PATTERN
7185  034222  012037  002206                MOV     (R0)+,T.MR2     ;       MR2
7186  034226  012037  002210                MOV     (R0)+,T.MR3     ;       MR3
7187  034232  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
7188  034234  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
7189  034236  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
7190  034240  000204                        RTS     R4
7191
7192                            .SBTTL  "FIRST SHALL BE LAST, LAST SHALL BE FIRST" SUBROUTINE
7193                            ;*      THE CONTENTS OF R3 IS SWAPPED END FOR END, I.E. BIT 15
7194                            ;*      BECOMES BIT 0 AND VICE VERSA,BIT 14 BECOMES BIT
7195                            ;*      1 AND VICE VERSA, ETC.
7196                            ;*
7197                            ;*      CALL:
7198                            ;*      JSR     R4,FSBLVV
7199                            ;*      WITH R3 LOADED WITH THE WORD TO BE SWAPPED
7200                            ;*
7201                            ;*      RETURN:
7202                            ;*      RTS     R4
7203                            ;*      WITH R3 SWAPPED.
7204
7205  034242                    FSBLVV:
7206  034242  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7207  034244  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7208  034246  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
7209  034250  005004                        CLR     R4              ;CLEAR R4 FOR SWAPPED WORD
7210  034252  012700  000001                MOV     #BIT0,R0        ;SET FOR BIT TEST
7211  034256  012701  100000                MOV     #BIT15,R1       ;SET FOR BIT SET
7212  034262  030003                1$:     BIT     R0,R3           ;TEST IF BIT SET
7213  034264  001401                        BEQ     2$              ;NO - SKIP
7214  034266  050104                        BIS     R1,R4           ;SET CORRESPONDING BIT
7215  034270  006300                2$:     ASL     R0              ;SHIFT FOR NEXT BIT TEST
7216  034272  001403                        BEQ     3$              ;LAST BIT TESTED - YES - EXIT
7217  034274  000241                        CLC                     ;CLEAR CARRY
7218  034276  006001                        ROR     R1              ;SHIFT FOR NEXT BIT SET
7219  034300  000770                        BR      1$              ;LOOP
7220  034302  010403                3$:     MOV     R4,R3           ;STORED SWAPPED WORD
7221  034304  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
7222  034306  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
7223  034310  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
7224  034312  000204                        RTS     R4
7225
7226
7227                            .SBTTL  CHECK FOR MEMORY CHECK ENABLE
7228
7229  034314  012737  034400  000004  PARCHK: MOV     #20$,ERRVEC     ;SET VECTOR FOR MEMORY PARITY CHECK
7230  034322  012737  000340  000006        MOV     #PR7,ERRVEC+2
7231  034330  012737  000000  002334        MOV     #0,BADPAR       ;LOAD GOOD PARITY
7232  034336  005037  002332                CLR     MEMPAR          ;CLEAR FLAG
7233  034342  012703  172100                MOV     #MEMBAS,R3      ;LOAD REGISTER TO DETERMINE IF
7234                                                                ;   MEMORY CHECK ENABLE AVAILIBLE
7235  034346  012704  000001                MOV     #1,R4           ;INITIALIZE MASK
7236  034352  012713  000001        16$:    MOV     #PAR.EN,(R3)    ;ENABLE MEMORY CHECK
```

```
7237  034356  005713                          TST     (R3)
7238  034360  050437  002332                  BIS     R4,MEMPAR       ;SET FLAG
7239  034364  062703  000002                  ADD     #2,R3
7240  034370  000241                          CLC
7241  034372  006104                          ROL     R4              ;CHECK IF FINISHED
7242  034374  001366                          BNE     16$             ;NO, SET UP NEXT MEMORY PARITY MODULE
7243  034376  000406                          BR      22$             ;RESTORE TRAP VECTOR
7244
7245  034400  022626                  20$:    CMP     (SP)+,(SP)+     ;ADJUST STACK
7246  034402  062703  000002                  ADD     #2,R3
7247  034406  000241                          CLC
7248  034410  006104                          ROL     R4              ;CHECK IF FINISHED
7249  034412  001357                          BNE     16$             ;NO, GET NEXT LOCATION
7250  034414  012737  000006  000004  22$:    MOV     #ERRVEC+2,ERRVEC ;RESTORE TRAP CATCHER
7251  034422  005037  000006                  CLR     ERRVEC+2
7252  034426  005737  002332                  TST     MEMPAR          ;CHECK IF MEMORY CHECK ENABLE AVAILIABLE
7253  034432  001005                          BNE     25$             ;YES, RETURN
7254  034434  012737  000116  000114          MOV     #MEMVEC+2,MEMVEC ;RESTORE TRAP CATCHER
7255  034442  005037  000116                  CLR     MEMVEC+2
7256  034446  000207                  25$:    RTS     PC              ;RETURN
7257                                  .SBTTL  ROUTINE TO SIZE MEMORY
7258
7259                                  ;;**************************************************************
7260                                  ;*CALL:
7261                                  ;*      JSR     PC,$SIZE
7262                                  ;*      RETURN
7263                                  ;*$LSTAD WILL CONTAIN:
7264                                  ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
7265                                  ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
7266                                  ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
7267                                  ;*$KT11 IS THE MEMORY MANAGEMENT KEY
7268                                  ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
7269                                  ;*      MUST BE SETUP BEFORE THE CALL
7270                                  ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
7271                                  ;*      DETERMINED BY ROUTINE
7272
7273  034450  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
7274  034452  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
7275  034454  010246                          MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
7276  034456  010346                          MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
7277  034460  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
7278  034464  013746  000006                  MOV     @#ERRVEC+2,-(SP)
7279  034470  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
7280                                  ;;SET THE ERRVEC PS TO THE PRESENT PS
7281  034472  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
7282  034474  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
7283  034500  012701  003776                  MOV     #3776,R1        ;;SETUP ADDRESS
7284  034504  105727                          TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
7285  034506  000200                  $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
7286  034510  100062                          BPL     $CORE           ;;BR IF NO
7287  034512  012737  034650  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
7288  034520  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
7289  034524  052737  100000  034506          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
7290  034532  005046                          CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
7291  034534  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
7292  034540  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
```

```
7293  034544  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
7294  034552  011622                          MOV     (SP),(R2)+      ;;LOAD "PAR"
7295  034554  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
7296  034560  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
7297  034562  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
7298  034566  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
7299  034570  012737  034606  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
7300  034576  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
7301  034604  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
7302  034606  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
7303  034610  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
7304  034614  012737  034640  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
7305  034622  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
7306  034626  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
7307  034632  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
7308  034636  101371                          BHI     4$              ;;NO--TRY IT
7309  034640  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
7310  034642  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
7311  034646  000421                          BR      $SIZEX
7312  034650  042737  100000  034506  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
7313  034656  012737  034706  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
7314  034664  005002                          CLR     R2              ;;SET UP BANK
7315  034666  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
7316  034672  062702  000040                  ADD     #40,R2          ;;1K STEP
7317  034676  005711                          TST     (R1)            ;;TRAP ON TIME OUT
7318  034700  022701  177776                  CMP     #177776,R1      ;;LAST ONE
7319  034704  001370                          BNE     1$              ;;NO--TRY AGAIN
7320  034706  162701  004000  $CROUT: SUB     #4000,R1
7321  034712  162702  000040  $SIZEX: SUB     #40,R2          ;;DROP BACK
7322  034716  010006                          MOV     R0,SP           ;;RESTORE THE STACK
7323  034720  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
7324  034724  012637  000004                  MOV     (SP)+,@#ERRVEC
7325  034730  010137  034752                  MOV     R1,$LSTAD       ;;LAST ADDRESS
7326  034734  010237  034754                  MOV     R2,$LSTBK       ;;LAST BANK
7327  034740  012603                          MOV     (SP)+,R3        ;;RESTORE R3
7328  034742  012602                          MOV     (SP)+,R2        ;;RESTORE R2
7329  034744  012601                          MOV     (SP)+,R1        ;;RESTORE R1
7330  034746  012600                          MOV     (SP)+,R0        ;;RESTORE R0
7331  034750  000207                          RTS     PC
7332  034752  000000  $LSTAD: .WORD   0       ;;CONTAINS THE LAST ADDRESS
7333  034754  000000  $LSTBK: .WORD   0       ;;CONTAINS THE LAST BANK
7334                    .SBTTL  SCOPE HANDLER ROUTINE
7335
7336                    ;;******************************************************************
7337                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7338                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7339                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7340                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7341                    ;*SW14=1         LOOP ON TEST
7342                    ;*SW11=1         INHIBIT ITERATIONS
7343                    ;*SW09=1         LOOP ON ERROR
7344                    ;*SW08=1         LOOP ON TEST IN SWR<7:0>
7345                    ;*CALL
7346                    ;*      SCOPE           ;;SCOPE=IOT
7347
7348  034756            $SCOPE:
```

```
7349  034756  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
7350  034760  032777  040000  144152  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
7351  034766  001131                          BNE     $OVER           ;;YES IF SW14=1
7352                                  ;#####START OF CODE FOR THE XOR TESTER#####
7353  034770  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
7354                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7355  034772  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7356  034776  012737  035016  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
7357  035004  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
7358  035010  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
7359  035014  000500                          BR      $SVLAD          ;;GO TO THE NEXT TEST
7360  035016  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
7361  035020  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
7362  035024  000440                          BR      7$              ;;LOOP ON THE PRESENT TEST
7363  035026                          6$:;#####END OF CODE FOR THE XOR TESTER#####
7364  035026  032777  000400  144104          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
7365  035034  001421                          BEQ     2$              ;;BR IF NO
7366  035036  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
7367  035040  117716  144074                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
7368  035044  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
7369  035046  022716  000042                  CMP     #42,(SP)        ;;CHECK THE NUMBER IN THE SWR
7370  035052  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
7371  035054  011637  001102                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
7372  035060  005316                          DEC     (SP)            ;;BACKUP BY ONE
7373  035062  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
7374  035064  062716  035270                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
7375  035070  013637  001106                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
7376  035074  000466                          BR      $OVER           ;;GO LOOP ON THE TEST
7377  035076  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
7378  035100  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
7379  035104  001421                          BEQ     3$              ;;BR IF NO
7380  035106  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
7381  035114  101015                          BHI     3$              ;;BR IF NO
7382  035116  032777  001000  144014          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
7383  035124  001404                          BEQ     4$              ;;BR IF NO
7384  035126  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
7385  035134  000446                          BR      $OVER
7386  035136  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
7387  035142  005037  001234                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
7388  035146  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
7389  035150  032777  004000  143762  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
7390  035156  001011                          BNE     1$              ;;BR IF YES
7391  035160  005737  001256                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
7392  035164  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
7393  035166  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
7394  035172  023737  001234  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
7395  035200  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
7396  035202  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
7397  035210  013737  035266  001234          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
7398  035216  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
7399  035222  113737  001102  001254          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
7400  035230  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
7401  035234  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
7402  035240  005037  001236                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
7403  035244  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7404  035252  013777  001102  143662  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
```

```
7405  035260  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
7406  035264  000002                          RTI                     ;;FIXES PS
7407  035266  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
7408  035270                          $SW08TBL:
7409  035270  003426                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
7410  035272  004040                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
7411  035274  004452                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
7412  035276  005064                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
7413  035300  005476                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
7414  035302  006110                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
7415  035304  006522                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
7416  035306  007144                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
7417  035310  007414                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
7418  035312  007656                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
7419  035314  010126                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
7420  035316  010370                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
7421  035320  010704                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
7422  035322  011176                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
7423  035324  011724                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
7424  035326  012452                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
7425  035330  013200                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
7426  035332  013726                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
7427  035334  014462                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
7428  035336  015216                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
7429  035340  015744                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
7430  035342  016260                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
7431  035344  016552                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
7432  035346  017300                          .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
7433  035350  020034                          .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
7434  035352  020350                          .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
7435  035354  021032                          .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
7436  035356  021346                          .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
7437  035360  022116                          .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
7438  035362  022426                          .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
7439  035364  022764                          .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
7440  035366  023314                          .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
7441  035370  023650                          .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
7442  035372  024224                          .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
7443                          ;;**************************************************************
7444                          .SBTTL  LOOP ON INTERNAL ERROR
7445
7446  035374  032777  001000  143536  SCOP1$: BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
7447  035402  001405                          BEQ     5$              ;NO RETURN
7448  035404  105737  001103                  TSTB    $ERFLG          ;CHECK IF ERROR OCCURED
7449  035410  001402                          BEQ     5$              ;NO, RETURN
7450  035412  013716  001110                  MOV     $LPERR,(SP)     ;GO BACK TO BEGINNING OF LOOP
7451  035416  000002                  5$:     RTI                     ;RETURN
7452                          .SBTTL  APT COMMUNICATIONS ROUTINE
7453
7454                          ;;**************************************************************
7455  035420  112737  000001  035664  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
7456  035426  112737  000001  035662  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
7457  035434  000403                          BR      $ATYC
7458  035436  112737  000001  035664  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
7459  035444                          $ATYC:
7460  035444  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

```
7461  035446  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7462  035450  105737  035662                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
7463  035454  001450                          BEQ     5$              ;;IF NOT:  BR
7464  035456  122737  000001  001270          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
7465  035464  001031                          BNE     3$              ;;IF NOT:  BR
7466  035466  132737  000100  001271          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
7467  035474  001425                          BEQ     3$              ;;IF NOT:  BR
7468  035476  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
7469  035502  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
7470  035510  005737  001250          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
7471  035514  001375                          BNE     1$              ;;IF NOT:  WAIT
7472  035516  010037  001264                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
7473  035522  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
7474  035524  001376                          BNE     2$
7475  035526  163700  001264                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
7476  035532  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
7477  035534  010037  001266                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
7478  035540  012737  000004  001250          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
7479  035546  000413                          BR      5$
7480  035550  017637  000004  035574  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
7481  035556  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
7482  035564  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
7483  035570  004737  036444                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
7484  035574  000000                  4$:     .WORD   0
7485  035576                          5$:
7486  035576  105737  035664          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
7487  035602  001416                          BEQ     12$             ;;IF NOT:  BR
7488  035604  005737  001270                  TST     $ENV            ;;RUNNING UNDER APT?
7489  035610  001413                          BEQ     12$             ;;IF NOT:  BR
7490  035612  005737  001250          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
7491  035616  001375                          BNE     11$             ;;IF NOT:  WAIT
7492  035620  017637  000004  001252          MOV     @4(SP),$FATAL   ;;GET ERROR #
7493  035626  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
7494  035634  005237  001250                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
7495  035640  105037  035664          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
7496  035644  105037  035663                  CLRB    $LFLG           ;;CLEAR LOG FLAG
7497  035650  105037  035662                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
7498  035654  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
7499  035656  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
7500  035660  000207                          RTS     PC              ;;RETURN
7501  035662     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
7502  035663     000          $LFLG:  .BYTE   0               ;;LOG FLAG
7503  035664     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
7504        035666                          .EVEN
7505        000200          APTSIZE=200
7506        000001          APTENV=001
7507        000100          APTSPOOL=100
7508        000040          APTCSUP=040
7509                        .SBTTL  ERROR HANDLER ROUTINE
7510
7511                        ;;*****************************************************************
7512                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7513                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7514                        ;*AND GO TO TYPERR ON ERROR
7515                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7516                        ;*SW15=1         HALT ON ERROR
```

```
                                        ;*SW13=1        INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1        BELL ON ERROR
                                        ;*SW09=1        LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

035666                                  $ERROR:
035666  104407                                  CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
035670  105237  001103                  7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
035674  001775                          1$:     BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
035676  013777  001102  143236                  MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
035704  032777  002000  143226                  BIT     #BIT10,@SWR             ;;BELL ON ERROR?
035712  001402                                  BEQ     1$                      ;;NO - SKIP
035714  104401  001240                          TYPE    ,$BELL                  ;;RING BELL
035720  005237  001112                  1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
035724  011637  001116                          MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
035730  162737  000002  001116                  SUB     #2,$ERRPC
035736  117737  143154  001114                  MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
035744  032777  020000  143166                  BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
035752  001004                                  BNE     20$                     ;;SKIP TYPEOUTS
035754  004737  036066                          JSR     PC,TYPERR               ;;GO TO USER ERROR ROUTINE
035760  104401  001245                          TYPE    ,$CRLF
035764                                  20$:
035764  122737  000001  001270                  CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
035772  001007                                  BNE     2$                      ;;NO,SKIP APT ERROR REPORT
035774  113737  001114  036006                  MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
036002  004737  035436                          JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
036006     000                  21$:    .BYTE   0
036007     000                          .BYTE   0
036010  000777                  22$:    BR      22$                     ;;APT ERROR LOOP
036012  005777  143122          2$:     TST     @SWR                    ;;HALT ON ERROR
036016  100002                          BPL     3$                      ;;SKIP IF CONTINUE
036020  000000                          HALT                            ;;HALT ON ERROR!
036022  104407                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
036024  032777  001000  143106  3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
036032  001402                          BEQ     4$                      ;;BR IF NO
036034  013716  001110                  MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
036040  005737  001236          4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
036044  001402                          BEQ     5$                      ;;BR IF NONE
036046  013716  001236                  MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
036052                                  5$:
036052  022737  025042  000042                  CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
036060  001001                          BNE     6$                      ;;BRANCH IF NO
036062  000000                          HALT                            ;;YES
036064                                  6$:
036064  000002                          RTI                             ;;RETURN

                                        ;;*****************************************************************
                                        .SBTTL  TYPE ERROR ROUTINE
                                        ;*ENTRY JSR PC,TYPERR
                                        ;*RETURN RTS PC
                                        ;*
                                        ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                        ;*ERROR IS TO BE REPORTED. IT THEN USES THE "ERROR TABLE" ($ERRTB)
                                        ;*ENTRY TO DEFINE WHAT INFORMATION IS TO BE REPORTED CONCERNING
                                        ;*THE ERROR.
```

```
                                ;;*****************************************************************
036066  104413                 TYPERR: SAVREG
036070  113737  001102  001254         MOVB    $TSTNM,$TESTN   ;SAVE TEST NUMBER FOR REPORT
036076  042737  177400  001254         BIC     #177400,$TESTN  ;CLEAR UNUSED BITS
036104  113700  001114                 MOVB    $ITEMB,R0       ;ENTER ERROR NUMBER
036110  042700  177400                 BIC     #177400,R0      ;CLEAR UNUSED BITS
036114  005300                         DEC     R0              ;FORM INDEX FOR ERROR TABLE
036116  006300                         ASL     R0
036120  006300                         ASL     R0
036122  006300                         ASL     R0
036124  062700  001330         1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS OF ERROR ENTRY
036130  012037  036144                 MOV     (R0)+,2$        ;GET EM POINTER
036134  001404                         BEQ     3$              ;BRANCH IF THERE ISN'T ONE
036136  104401  001245                 TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN LINE FEED
036142  104401                         TYPE                    ;TYPE ERROR MESSAGE (EM)
036144  000000         2$:     .WORD   0               ;EM POINTER GOES HERE
036146  012037  036162         3$:     MOV     (R0)+,4$        ;GET DH POINTER
036152  001404                         BEQ     5$              ;BRANCH IF THERE ISN'T ONE
036154  104401  001245                 TYPE    ,$CRLF          ;TYPE CR-LF
036160  104401                         TYPE                    ;TYPE DATA HEADER
036162  000000         4$:     .WORD   0               ;DH POINTER GOES HERE
036164  012001         5$:     MOV     (R0)+,R1        ;GET DT POINTER
036166  001445                         BEQ     20$             ;BRANCH IF THERE ARE NONE
036170  005004                         CLR     R4              ;RESET INDENT SWITCH
036172  012000                         MOV     (R0)+,R0        ;GET DF POINTER
036174  012002                         MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
036176  104401  001245                 TYPE    ,$CRLF
036202  112003         10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
036204  105720                         TSTB    (R0)+           ;BUMP PAST FORMAT WORD
036206  005703                         TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
036210  001416                         BEQ     14$             ;NO - SKIP DATA PRINT
036212  005704                         TST     R4              ;CHECK IF INDENT WORDS
036214  001004                         BNE     12$             ;YES, GO INDENT
036216  013146         11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
036220  104402                         TYPOC                   ;TYPE IT
036222  005303                         DEC     R3              ;MORE DATA WORDS
036224  001403                         BEQ     13$             ;NO-BRANCH
036226  104401  042724         12$:    TYPE    ,SPACE2         ;TYPE SEPARATORS
036232  000771                         BR      11$             ;LOOP
036234  104401  001245         13$:    TYPE    ,$CRLF          ;TYPE <CR><LF>
036240  005710                         TST     (R0)            ;CHECK IF NEXT HEADER AVAILIBLE
036242  001401                         BEQ     14$             ;NO, DO NOT CHANGE INDENT
036244  005104                         COM     R4              ;CHANGE INDENT
036246  005302         14$:    DEC     R2              ;MORE DH'S?
036250  003414                         BLE     20$             ;NO-BRANCH
036252  012037  036272         15$:    MOV     (R0)+,18$       ;GET NEXT DH POINTER
036256  001751                         BEQ     10$             ;IF NO HEADER GET DATA
036260  005704                         TST     R4              ;INDENT?
036262  001402                         BEQ     17$             ;NO-BRANCH
036264  104401  042724                 TYPE    ,SPACE2         ;INDENT
036270  104401         17$:    TYPE                    ;TYPE DH
036272  000000         18$:    .WORD   0               ;DH POINTER GOES HERE
036274  104401  001245                 TYPE    ,$CRLF
036300  000740                         BR      10$             ;LOOP
036302  104414         20$:    RESREG
036304  005237  002314                 INC     ERRCNT          ;INCREMENT ERROR COUNT
```

```
7629  036310  032777  010000  142622          BIT     #SW12,@SWR      ;CHECK IF ABORT AFTER 20 ERRORS
7630  036316  001421                          BEQ     25$             ;NO, RETURN
7631  036320  022737  000024  002314          CMP     #20.,ERRCNT     ;CHECK IF EROR THRESHOLD EXCEEDED
7632  036326  103015                          BHIS    25$             ;NO, RETURN
7633  036330  104401  042727                  TYPE    ,ABORT          ;TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
7634                                                                  ;  ERROR THRESHOLD EXCEEDED"
7635  036334  005737  000042                  TST     42              ;CHECK IF IN CHAIN MODE
7636  036340  001407                          BEQ     30$             ;NO, HALT
7637  036342  012737  000001  024700          MOV     #1,$EOPCT       ;FORCE END OF PASS COUNT TO ONE FOR ABORT
7638  036350  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK
7639  036354  000137  024652                  JMP     $EOP            ;BRING IN NEXT PROGRAM IN CHAIN
7640  036360  000000                  30$:    HALT
7641  036362  000207                  25$:    RTS     PC
7642
7643                                  .SBTTL  CONTROLLED PROGRAM HALT
7644
7645  036364  013702  001324          CTRHLT: MOV     $BASE,R2        ;SET '611 BASE
7646  036370  012762  005000  000010          MOV     #CLR,RKCS2(R2)  ;CLEAR SUBSYSTEM
7647  036376  005037  001236                  CLR     $ESCAPE         ;CLEAR ESCAPE
7648  036402  105037  001103                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
7649  036406  012706  001100                  MOV     #STACK,SP       ;CLEAR STACK
7650  036412  104401  042661                  TYPE    ,OPR007         ;TYPE HALT MESSAGE
7651  036416  005737  000042                  TST     42              ;TEST IF MONITOR PRESENT
7652  036422  001405                          BEQ     5$              ;NO SKIP
7653  036424  012737  000001  024700          MOV     #1,$EOPCT       ;FORCE END OF PROGRAM
7654  036432  000137  024652                  JMP     $EOP            ;JUMP TO END OF PASS
7655
7656  036436  000000                  5$:     HALT                    ;HALT PROGRAM
7657  036440  000137  002404                  JMP     START1          ;RESTART IF CONTINUE
7658
7659                                  .SBTTL  TYPE ROUTINE
7660
7661                                  ;;*****************************************************************
7662                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7663                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
7664                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7665                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7666                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7667                                  ;*
7668                                  ;*CALL:
7669                                  ;*1) USING A TRAP INSTRUCTION
7670                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7671                                  ;*OR
7672                                  ;*      TYPE
7673                                  ;*      MESADR
7674                                  ;*
7675
7676  036444  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
7677  036450  100002                          BPL     1$              ;;BR IF YES
7678  036452  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
7679  036454  000430                          BR      3$              ;;LEAVE
7680  036456  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
7681  036460  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
7682  036464  122737  000001  001270          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
7683  036472  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
7684  036474  132737  000100  001271          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
 7685  036502  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
 7686  036504  010037  036514                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 7687  036510  004737  035426                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 7688  036514  000000              61$:      .WORD   0               ;;MESSAGE ADDRESS
 7689  036516  132737  000040  001271 62$:   BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 7690  036524  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
 7691  036526  112046              2$:       MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 7692  036530  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 7693  036532  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 7694  036534  012600              60$:      MOV     (SP)+,R0        ;;RESTORE R0
 7695  036536  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 7696  036542  000002                        RTI                     ;;RETURN
 7697  036544  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 7698  036550  001430                        BEQ     8$
 7699  036552  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 7700  036556  001006                        BNE     5$
 7701  036560  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
 7702  036562  104401                        TYPE                    ;;TYPE A CR AND LF
 7703  036564  001245                        $CRLF
 7704  036566  105037  036722                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 7705  036572  000755                        BR      2$              ;;GET NEXT CHARACTER
 7706  036574  004737  036656        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 7707  036600  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 7708  036604  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
 7709  036606  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 7710                                                                ;;AND THE NULL CHAR.
 7711  036612  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 7712  036616  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 7713  036620  004737  036656                JSR     PC,$TYPEC       ;;GO TYPE A NULL
 7714  036624  105337  036722                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 7715  036630  000770                        BR      7$              ;;LOOP
 7716
 7717                                  ;HORIZONTAL TAB PROCESSOR
 7718
 7719  036632  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 7720  036636  004737  036656        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 7721  036642  132737  000007  036722        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 7722  036650  001372                        BNE     9$              ;;TAB STOP
 7723  036652  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
 7724  036654  000724                        BR      2$              ;;GET NEXT CHARACTER
 7725  036656  105777  142266        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 7726  036662  100375                        BPL     $TYPEC
 7727  036664  116677  000002  142260        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 7728  036672  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 7729  036700  001003                        BNE     1$              ;;BRANCH IF NO
 7730  036702  105037  036722                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 7731  036706  000406                        BR      $TYPEX          ;;EXIT
 7732  036710  122766  000012  000002 1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 7733  036716  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
 7734  036720  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
 7735  036722  000000        $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
 7736  036724  000207        $TYPEX: RTS     PC
 7737
 7738                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 7739
 7740                                  ;;**************************************************************
```

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

036726  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
036732  116637  000001  037151          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
036740  112637  037153                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
036744  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
036750  000406                          BR      $TYPON
036752  112737  000001  037151  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
036760  112737  000006  037153          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
036766  112737  000005  037150  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
036774  010346                          MOV     R3,-(SP)        ;;SAVE R3
036776  010446                          MOV     R4,-(SP)        ;;SAVE R4
037000  010546                          MOV     R5,-(SP)        ;;SAVE R5
037002  113704  037153                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
037006  005404                          NEG     R4
037010  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
037014  110437  037152                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
037020  113704  037151                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
037024  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
037030  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
037032  006105          1$:             ROL     R5              ;;ROTATE MSB INTO "C"
037034  000404                          BR      3$              ;;GO DO MSB
037036  006105          2$:             ROL     R5              ;;FORM THIS DIGIT
037040  006105                          ROL     R5
037042  006105                          ROL     R5
037044  010503                          MOV     R5,R3
037046  006103          3$:             ROL     R3              ;;GET LSB OF THIS DIGIT
037050  105337  037152                  DECB    $OMODE          ;;TYPE THIS DIGIT?
037054  100016                          BPL     7$              ;;BR IF NO
037056  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
037062  001002                          BNE     4$              ;;TEST FOR 0
037064  005704                          TST     R4              ;;SUPPRESS THIS 0?
037066  001403                          BEQ     5$              ;;BR IF YES
037070  005204          4$:             INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
037072  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
037076  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

```
7797  037102  110337  037146                  MOVB    R3,8$           ;;SAVE FOR TYPING
7798  037106  104401  037146                  TYPE    8$              ;;GO TYPE THIS DIGIT
7799  037112  105337  037150          7$:     DECB    $OCNT           ;;COUNT BY 1
7800  037116  003347                          BGT     2$              ;;BR IF MORE TO DO
7801  037120  002402                          BLT     6$              ;;BR IF DONE
7802  037122  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
7803  037124  000744                          BR      2$              ;;GO DO THE LAST DIGIT
7804  037126  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
7805  037130  012604                          MOV     (SP)+,R4        ;;RESTORE R4
7806  037132  012603                          MOV     (SP)+,R3        ;;RESTORE R3
7807  037134  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
7808  037142  012616                          MOV     (SP)+,(SP)
7809  037144  000002                          RTI                     ;;RETURN
7810  037146     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
7811  037147     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
7812  037150     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
7813  037151     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
7814  037152  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
7815                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
7816
7817                                  ;;**************************************************************
7818                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
7819                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
7820                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
7821                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
7822                                  ;*REPLACED WITH SPACES.
7823                                  ;*CALL:
7824                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
7825                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
7826
7827  037154                          $TYPDS:
7828  037154  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
7829  037156  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
7830  037160  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
7831  037162  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
7832  037164  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
7833  037166  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
7834  037172  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
7835  037176  100004                          BPL     1$              ;;BR IF INPUT IS POS.
7836  037200  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
7837  037202  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
7838  037210  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
7839  037212  012703  037370                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
7840  037216  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
7841  037222  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
7842  037224  016001  037360                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
7843  037230  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
7844  037232  002402                          BLT     4$              ;;BR IF DONE
7845  037234  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
7846  037236  000774                          BR      3$
7847  037240  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
7848  037242  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
7849  037244  001002                          BNE     5$              ;;FALL THROUGH IF 0
7850  037246  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
7851  037250  100407                          BMI     7$              ;;BR IF YES
7852  037252  106316                  5$:     ASLB    (SP)            ;;MSD?
```

```
7853  037254  103003                        BCC     6$                      ;;BR IF NO
7854  037256  116663  000001  177777        MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
7855  037264  052702  000060          6$:   BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
7856  037270  052702  000040          7$:   BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7857  037274  110223                        MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7858  037276  005720                        TST     (R0)+                   ;;JUST INCREMENTING
7859  037300  020027  000010                CMP     R0,#10                  ;;CHECK THE TABLE INDEX
7860  037304  002746                        BLT     2$                      ;;GO DO THE NEXT DIGIT
7861  037306  003002                        BGT     8$                      ;;GO TO EXIT
7862  037310  010502                        MOV     R5,R2                   ;;GET THE LSD
7863  037312  000764                        BR      6$                      ;;GO CHANGE TO ASCII
7864  037314  105726                  8$:   TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
7865  037316  100003                        BPL     9$                      ;;BR IF NO
7866  037320  116663  177777  177776        MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
7867  037326  105013                  9$:   CLRB    (R3)                    ;;SET THE TERMINATOR
7868  037330  012605                        MOV     (SP)+,R5                ;;POP STACK INTO R5
7869  037332  012603                        MOV     (SP)+,R3                ;;POP STACK INTO R3
7870  037334  012602                        MOV     (SP)+,R2                ;;POP STACK INTO R2
7871  037336  012601                        MOV     (SP)+,R1                ;;POP STACK INTO R1
7872  037340  012600                        MOV     (SP)+,R0                ;;POP STACK INTO R0
7873  037342  104401  037370                TYPE    ,$DBLK                  ;;NOW TYPE THE NUMBER
7874  037346  016666  000002  000004        MOV     2(SP),4(SP)             ;;ADJUST THE STACK
7875  037354  012616                        MOV     (SP)+,(SP)
7876  037356  000002                        RTI                             ;;RETURN TO USER
7877  037360  023420                  $DTBL:  10000.
7878  037362  001750                         1000.
7879  037364  000144                         100.
7880  037366  000012                         10.
7881  037370  000004                  $DBLK:  .BLKW   4
7882                                  .SBTTL  TTY INPUT ROUTINE
7883
7884                                  ;;*****************************************************************
7885                                  .ENABL  LSB
7886  037400  000000                  $TKCNT: .WORD   0                       ;;NUMBER OF ITEMS IN QUEUE
7887  037402  000000                  $TKQIN: .WORD   0                       ;;INPUT POINTER
7888  037404  000000                  $TKQOUT: .WORD  0                       ;;OUTPUT POINTER
7889  037406  000001                  $TKQSRT: .BLKB  1                       ;;TTY KEYBOARD QUEUE
7890          037407                  $TKQEND=.
7891          037410                  .EVEN
7892
7893                                  ;*TK INITIALIZE ROUTINE
7894                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
7895                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
7896                                  ;
7897                                  ;*CALL:
7898                                  ;*      JSR     PC,$TKINT
7899                                  ;*      RETURN
7900                                  ;
7901  037410  005037  037400          $TKINT: CLR     $TKCNT                  ;;CLEAR COUNT OF ITEMS IN QUEUE
7902  037414  012737  037406  037402        MOV     #$TKQSRT,$TKQIN         ;;MOVE THE STARTING ADDRESS OF THE
7903  037422  013737  037402  037404        MOV     $TKQIN,$TKQOUT          ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
7904  037430  012737  037460  000060        MOV     #$TKSRV,@#TKVEC         ;;INITIALIZE THE KEYBOARD VECTOR
7905  037436  012737  000200  000062        MOV     #200,@#TKVEC+2          ;;"BR" LEVEL 4
7906  037444  005777  141476                TST     @$TKB                   ;;CLEAR DONE FLAG
7907  037450  012777  000100  141466        MOV     #100,@$TKS              ;;ENABLE TTY KEYBOARD INTERRUPT
7908  037456  000207                        RTS     PC                      ;;RETURN TO CALLER
```

```
7909
7910                                    ;*TK SERVICE ROUTINE
7911                                    ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
7912                                    ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
7913                                    ;*IT IN THE QUEUE.
7914                                    ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
7915                                    ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
7916                                    ;
7917  037460  117746  141462           $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
7918  037464  042716  177600                   BIC     #↑C177,(SP)     ;;STRIP THE JUNK
7919  037470  021627  000003                   CMP     (SP),#3         ;;IS IT A CONTROL C?
7920  037474  001007                           BNE     1$              ;;BRANCH IF NO
7921  037476  104401  040574                   TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
7922  037502  004737  037410                   JSR     PC,$TKINT       ;;INIT THE KEYBOARD
7923  037506  005726                           TST     (SP)+           ;;CLEAN UP STACK
7924  037510  000137  036364                   JMP     CTRHLT          ;;CONTROL C RESTART
7925  037514  021627  000007           1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
7926  037520  001004                           BNE     2$              ;;BRANCH IF NO
7927  037522  022737  000176  001140           CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
7928  037530  001500                           BEQ     6$              ;;GO TO SWR CHANGE
7929
7930  037532                           2$:
7931  037532  022737  000001  037400           CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
7932  037540  001004                           BNE     3$              ;;BRANCH IF NO
7933  037542  104401  001240                   TYPE    ,$BELL          ;;RING THE TTY BELL
7934  037546  005726                           TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
7935  037550  000451                           BR      5$              ;;EXIT
7936  037552  021627  000023           3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
7937  037556  001021                           BNE     32$             ;;BRANCH IF NO
7938  037560  005077  141360                   CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
7939  037564  005726                           TST     (SP)+           ;;CLEAN CHAR OFF STACK
7940  037566  105777  141352           31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
7941  037572  100375                           BPL     31$             ;;LOOP UNTIL ITS THERE
7942  037574  117746  141346                   MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
7943  037600  042716  177600                   BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
7944  037604  022627  000021                   CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
7945  037610  001366                           BNE     31$             ;;BRANCH IF NO
7946  037612  012777  000100  141324           MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
7947  037620  000002                           RTI                     ;;RETURN
7948  037622  005237  037400           32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
7949  037626  021627  000140                   CMP     (SP),#140       ;;IS IT UPPER CASE?
7950  037632  002405                           BLT     4$              ;;BRANCH IF YES
7951  037634  021627  000175                   CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
7952  037640  003002                           BGT     4$              ;;BRANCH IF YES
7953  037642  042716  000040                   BIC     #40,(SP)        ;;MAKE IT UPPER CASE
7954  037646  112677  177530           4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
7955  037652  005237  037402                   INC     $TKQIN          ;;UPDATE THE POINTER
7956  037656  023727  037402  037407           CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
7957  037664  001003                           BNE     5$              ;;BRANCH IF NO
7958  037666  012737  037406  037402           MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
7959  037674  000002                   5$:     RTI                     ;;RETURN
7960
7961                                    ;;*****************************************************************
7962                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7963                                    ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
7964                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
```

```
7965                                            ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
7966  037676  022737  000176  001140  $CKSWR:  CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
7967  037704  001124                            BNE     15$             ;;EXIT IF NOT
7968  037706  105777  141232                    TSTB    @$TKS           ;;IS A CHAR WAITING?
7969  037712  100121                            BPL     15$             ;;IF NOT, EXIT
7970  037714  117746  141226                    MOVB    @$TKB,-(SP)     ;;YES
7971  037720  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
7972  037724  021627  000007                    CMP     (SP),#7         ;;IS IT A CONTROL-G?
7973  037730  001300                            BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
7974                                                                    ;;AND EXIT
7975
7976                                          ;;**************************************************************
7977                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
7978                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
7979                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
7980  037732  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
7981  037740  001674                            BEQ     2$              ;;BRANCH IF YES
7982  037742  005726                            TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
7983  037744  004737  037410                    JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
7984  037750  005077  141170                    CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
7985  037754  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
7986
7987  037762  104401  040606                    TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
7988  037766  104401  040613  $GTSWR:  TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
7989  037772  013746  000176                    MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
7990  037776  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7991  040000  104401  040624                    TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
7992  040004  005046                  19$:     CLR     -(SP)           ;;CLEAR COUNTER
7993  040006  005046                            CLR     -(SP)           ;;THE NEW SWR
7994  040010  105777  141130          7$:     TSTB    @$TKS           ;;CHAR THERE?
7995  040014  100375                            BPL     7$              ;;IF NOT TRY AGAIN
7996
7997  040016  117746  141124                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
7998  040022  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
7999
8000  040026  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL-C?
8001  040032  001015                            BNE     9$              ;;BRANCH IF NOT
8002  040034  104401  040574                    TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
8003  040040  062706  000006                    ADD     #6,SP           ;;CLEAN UP STACK
8004  040044  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
8005  040052  001003                            BNE     8$              ;;BRANCH IF NO
8006  040054  012777  000100  141062          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
8007  040062  000137  036364          8$:     JMP     CTRHLT          ;;CONTROL-C RESTART
8008
8009
8010  040066  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
8011  040072  001005                            BNE     10$             ;;BRANCH IF NOT
8012  040074  104401  040601                    TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
8013  040100  062706  000006          20$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
8014  040104  000737                            BR      19$             ;;LET'S TRY IT AGAIN
8015
8016
8017  040106  021627  000015          10$:     CMP     (SP),#15        ;;IS IT A <CR>?
8018  040112  001022                            BNE     16$             ;;BRANCH IF NO
8019  040114  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
8020  040120  001403                            BEQ     11$             ;;BRANCH IF YES
```

```
8021  040122  016677  000002  141010          MOV     2(SP),@SWR      ;;SAVE NEW SWR
8022  040130  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
8023  040134  104401  001245          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
8024  040140  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
8025  040146  001003                          BNE     15$             ;;BRANCH IF NOT
8026  040150  012777  000100  140766          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
8027  040156  000002                  15$:    RTI                     ;;RETURN
8028  040160  004737  036656          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
8029  040164  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
8030  040170  002420                          BLT     18$             ;;BRANCH IF YES
8031  040172  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
8032  040176  003015                          BGT     18$             ;;BRANCH IF YES
8033  040200  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
8034  040204  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
8035  040210  001403                          BEQ     17$             ;;BRANCH IF YES
8036  040212  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
8037  040214  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
8038  040216  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
8039  040220  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
8040  040224  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
8041  040230  000667                          BR      7$              ;;GET THE NEXT ONE
8042  040232  104401  001244          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
8043  040236  000720                          BR      20$             ;;SIMULATE CONTROL-U
8044                                  .DSABL  LSB
8045
8046
8047                                  ;;**************************************************************
8048                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
8049                                  ;*CALL:
8050                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
8051                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
8052                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
8053                                  ;
8054
8055  040240  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
8056  040242  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
8057  040250  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
8058  040254  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
8059  040256  012746  040264                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
8060  040262  000002                          RTI                     ;;POP NEW PC AND PS
8061  040264                          64$:
8062  040264  005737  037400          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
8063  040270  001775                          BEQ     1$
8064  040272  005337  037400                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
8065  040276  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
8066  040304  005237  037404                  INC     $TKQOUT         ;;UPDATE THE POINTER
8067  040310  023727  037404  037407          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
8068  040316  001003                          BNE     2$              ;;BRANCH IF NO
8069  040320  012737  037406  037404          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
8070  040326  000002                  2$:     RTI                     ;;RETURN
8071                                  ;;**************************************************************
8072                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
8073                                  ;*CALL:
8074                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
8075                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8076                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
```

```
8077
8078  040330  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
8079  040332  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
8080  040334  012703  040564            1$:   MOV     #$TTYIN,R3      ;;GET ADDRESS
8081  040340  022703  040574            2$:   CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
8082  040344  101456                          BLOS    4$              ;;BR IF YES
8083  040346  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
8084  040350  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
8085  040352  122713  000177            10$:  CMPB    #177,(R3)       ;;IS IT A RUBOUT
8086  040356  001022                          BNE     5$              ;;BR IF NO
8087  040360  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
8088  040362  001007                          BNE     6$              ;;BR IF NO
8089  040364  112737  000134  040562        MOVB    #'\,9$          ;;TYPE A BACK SLASH
8090  040372  104401  040562                TYPE    ,9$
8091  040376  012716  177777                MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
8092  040402  005303                  6$:   DEC     R3              ;;BACKUP BY ONE
8093  040404  020327  040564                CMP     R3,#$TTYIN      ;;STACK EMPTY?
8094  040410  103434                          BLO     4$              ;;BR IF YES
8095  040412  111337  040562                MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
8096  040416  104401  040562                TYPE    ,9$             ;;GO TYPE
8097  040422  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
8098  040424  005716                  5$:   TST     (SP)            ;;RUBOUT KEY SET?
8099  040426  001406                          BEQ     7$              ;;BR IF NO
8100  040430  112737  000134  040562        MOVB    #'\,9$          ;;TYPE A BACK SLASH
8101  040436  104401  040562                TYPE    ,9$
8102  040442  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
8103  040444  122713  000025            7$:   CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
8104  040450  001003                          BNE     8$              ;;BR IF NO
8105  040452  104401  040601                TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
8106  040456  000726                          BR      1$              ;;GO START OVER
8107  040460  122713  000022            8$:   CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
8108  040464  001011                          BNE     3$              ;;BRANCH IF NO
8109  040466  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
8110  040470  104401  001245                TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
8111  040474  104401  040564                TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
8112  040500  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
8113  040502  104401  001244            4$:   TYPE    ,$QUES          ;;TYPE A '?'
8114  040506  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
8115  040510  111337  040562            3$:   MOVB    (R3),9$         ;;ECHO THE CHARACTER
8116  040514  104401  040562                TYPE    ,9$
8117  040520  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
8118  040524  001305                          BNE     2$              ;;LOOP IF NOT RETURN
8119  040526  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
8120  040532  104401  001246                TYPE    ,$LF            ;;TYPE A LINE FEED
8121  040536  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
8122  040540  012603                          MOV     (SP)+,R3        ;;RESTORE R3
8123  040542  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
8124  040544  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
8125  040552  012766  040564  000004        MOV     #$TTYIN,4(SP)
8126  040560  000002                          RTI                     ;;RETURN
8127  040562     000                  9$:   .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
8128  040563     000                        .BYTE   0               ;;TERMINATOR
8129  040564  000010            $TTYIN: .BLKB   8.                  ;;RESERVE 8 BYTES FOR TTY INPUT
8130  040574  041536  005015     000    $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
8131  040601     136  006525  000012    $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
8132  040606  043536  005015     000    $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
```

```
8133  040613  015     051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
8134  040620  036440  000040
8135  040624  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
8136  040632  020075  000
8137          040636                  .EVEN
8138                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
8139
8140                                  ;;*****************************************************************
8141                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
8142                                  ;*CHANGE IT TO BINARY.
8143                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
8144                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
8145                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
8146                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
8147                                  ;*CALL:
8148                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
8149                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
8150                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
8151
8152  040636  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
8153  040640  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
8154  040646  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
8155  040650  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
8156  040652  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
8157  040654  104411          1$:     RDLIN                   ;;READ AN ASCIZ LINE
8158  040656  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
8159  040660  010037  040764          MOV     R0,5$           ;;AND SAVE IT
8160  040664  005001                  CLR     R1              ;;CLEAR DATA WORD
8161  040666  005002                  CLR     R2
8162  040670  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
8163  040672  001420                  BEQ     3$              ;;IF ZERO GET OUT
8164  040674  122716  000060          CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
8165  040700  003026                  BGT     4$              ;;IS AN OCTAL DIGIT
8166  040702  122716  000067          CMPB    #'7,(SP)
8167  040706  002423                  BLT     4$
8168  040710  006301                  ASL     R1              ;;*2
8169  040712  006102                  ROL     R2
8170  040714  006301                  ASL     R1              ;;*4
8171  040716  006102                  ROL     R2
8172  040720  006301                  ASL     R1              ;;*8
8173  040722  006102                  ROL     R2
8174  040724  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
8175  040730  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
8176  040732  000756                  BR      2$              ;;LOOP
8177  040734  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
8178  040736  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
8179  040742  010237  040774          MOV     R2,$HIOCT
8180  040746  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
8181  040750  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
8182  040752  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
8183  040754  000002                  RTI                     ;;RETURN
8184  040756  005726          4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
8185  040760  105010                  CLRB    (R0)            ;;SET A TERMINATOR
8186  040762  104401                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
8187  040764  000000          5$:     .WORD   0
8188  040766  104401  001244          TYPE    ,$QUES          ;;"?" "CR" & "LF"
```

```
                        BR      1$              ;;TRY AGAIN
$HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
.SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

;;**********************************************************************
;*SAVE R0-R5
;*CALL:
;*      SAVREG
;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
;*
;*TOP---(+16)
;* +2---(+18)
;* +4---R5
;* +6---R4
;* +8---R3
;*+10---R2
;*+12---R1
;*+14---R0

$SAVREG:
        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
        MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
        MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
        MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
        MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
        RTI

;*RESTORE R0-R5
;*CALL:
;*      RESREG
$RESREG:
        MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
        MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
        MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
        MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
        MOV     (SP)+,R5        ;;POP STACK INTO R5
        MOV     (SP)+,R4        ;;POP STACK INTO R4
        MOV     (SP)+,R3        ;;POP STACK INTO R3
        MOV     (SP)+,R2        ;;POP STACK INTO R2
        MOV     (SP)+,R1        ;;POP STACK INTO R1
        MOV     (SP)+,R0        ;;POP STACK INTO R0
        RTI

.SBTTL  POWER DOWN AND UP ROUTINE

;;**********************************************************************

;POWER DOWN ROUTINE
$PWRDN: MOV     @SWR,SAVSWR     ;SAVE SWITCH REGISTER
        MOV     #$PWRUP,PWRVEC  ;SET UP VECTOR
        MOV     #PR7,PWRVEC+2
```

```
8245  041114  000000                          HALT
8246  041116  000776                          BR      .-2     ;HANG UP
8247
8248                                  ;;*****************************************************************
8249
8250                                  ;POWER UP ROUTINE
8251  041120  005037  041214         $PWRUP: CLR     $PWRCT          ;LOAD WAIT COUNT
8252  041124  012737  000144  041216         MOV     #100.,$PWRCT+2
8253  041132  005237  041214         1$:    INC     $PWRCT          ;WAIT FOR TELETYPE
8254  041136  001375                         BNE     1$
8255  041140  005337  041216                 DEC     $PWRCT+2
8256  041144  001372                         BNE     1$
8257  041146  012737  041072  000024         MOV     #$PWRDN,PWRVEC  ;SET UP FOR POWER DOWN VECTOR
8258  041154  012737  000340  000026         MOV     #PR7,PWRVEC+2
8259  041162  012706  001100                 MOV     #STACK,SP       ;FORCE STACK
8260  041166  104401  041220                 TYPE    ,$POWER         ;TYPE POWER
8261  041172  004737  034314                 JSR     PC,PARCHK       ;REINITIALIZE MEMORY CHECK ENABLE
8262  041176  013777  002356  137734         MOV     SAVSWR,@SWR     ;RESTORE SWITCH REGISTER
8263  041204  013702  001324                 MOV     $BASE,R2        ;REINITIALISE R2 FOR '611 BASE
8264  041210  000177  137672                 JMP     @$LPADR ;GO BACK TO LAST TEST
8265
8266  041214  000000  000000         $PWRCT: .WORD   0,0             ;TELETYPE TIME OUT
8267  041220  047520  042527  000122 $POWER: .ASCIZ  /POWER/
8268                                         .EVEN
8269                                  .SBTTL  TRAP DECODER
8270
8271                                  ;;*****************************************************************
8272                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
8273                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
8274                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
8275                                  ;*GO TO THAT ROUTINE.
8276
8277  041226  010046                 $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
8278  041230  016600  000002                 MOV     2(SP),R0        ;;GET TRAP ADDRESS
8279  041234  005740                         TST     -(R0)           ;;BACKUP BY 2
8280  041236  111000                         MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
8281  041240  006300                         ASL     R0              ;;POSITION FOR INDEXING
8282  041242  016000  041262                 MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
8283  041246  000200                         RTS     R0              ;;GO TO ROUTINE
8284
8285
8286                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
8287
8288  041250  011646                 $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
8289  041252  016666  000004  000002         MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
8290  041260  000002                         RTI                     ;;RESTORE THE PSW
8291
8292                                  .SBTTL  TRAP TABLE
8293
8294                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
8295                                  ;*BY THE "TRAP" INSTRUCTION.
8296
8297                                  ;       ROUTINE
8298                                  ;       -------
8299  041262  041250                 $TRPAD: .WORD   $TRAP2
8300  041264  036444                         $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
```

```
 8301  041266  036752                $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 8302  041270  036726                $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 8303  041272  036766                $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 8304  041274  037154                $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
 8305
 8306  041276  037766                $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
 8307
 8308  041300  037676                $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
 8309  041302  040240                $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 8310  041304  040330                $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 8311  041306  040636                $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
 8312  041310  040776                $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
 8313  041312  041034                $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
 8314  041314  035374                SCOP1$  ;;CALL=SCOP1    TRAP+15(104415) INTERNAL LOOP ON ERROR
```

```
                                        .SBTTL  DATA TABLE FOR PRINT OUT
                                
041316  001254  002310                DT000:  .WORD   $TESTN,TRAPPC
041322  001254  001116  002220        DT0004: .WORD   $TESTN,$ERRPC,E.CS1,T.CS1,T.CS2,T.ER,T.DS
041330  002160  002170  002174
041336  002172
041340  001254  001116  002244        DT0005: .WORD   $TESTN,$ERRPC,E.MR1,T.MR1
041346  002204
041350  001254  001116  002246        DT0006: .WORD   $TESTN,$ERRPC,E.MR2,T.MR2
041356  002206
041360  001254  001116  002250        DT0007: .WORD   $TESTN,$ERRPC,E.MR3,T.MR3
041366  002210
041370  001254  001116  002244        DT0010: .WORD   $TESTN,$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
041376  002204  002320  002322
041404  002326
041406  001254  001116  002220        DT0011: .WORD   $TESTN,$ERRPC,E.CS1,T.CS1,PR.BIT,M1.BIT,BITCNT,T.CS2,T.ER,T.DS
041414  002160  002320  002322
041422  002326  002170  002174
041430  002172
041432  001254  001116  002240        DT0014: .WORD   $TESTN,$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
041440  002200  002226  002166
041446  001254  001116  002244        DT0017: .WORD   $TESTN,$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT
041454  002204  002316  002320
041462  002322  002324  002326
041470  001254  001116  002254        DT0021: .WORD   $TESTN,$ERRPC,E.ECPT,T.ECPT,BITCNT
041476  002214  002326
041502  001254  001116  002160        DT0026: .WORD   $TESTN,$ERRPC,T.CS1,T.CS2,T.ER,T.DS,T.WC,T.DCYL,T.DA
041510  002170  002174  002172
041516  002162  002200  002166
041524  001254  001116                DT0037: .WORD   $TESTN,$ERRPC
041530  001162  001164  001166        DT0040: .WORD   $REG0,$REG1,$REG2
041536  001254  001116  002254        DT0041: .WORD   $TESTN,$ERRPC,E.ECPT,T.ECPT,E.ECPS,T.ECPS,BITCNT
041544  002214  002252  002212
041552  002326
041554  001254  001116  002252        DT0043: .WORD   $TESTN,$ERRPC,E.ECPS,T.ECPS,BITCNT
041562  002212  002326
041566  001254  001116  002234        DT0045: .WORD   $TESTN,$ERRPC,E.ER,T.ER
041574  002174
041576  001254  001116  002224        DT0046: .WORD   $TESTN,$ERRPC,E.BA,T.BA
041604  002164
041606  001254  001116  002222        DT0062: .WORD   $TESTN,$ERRPC,E.WC,T.WC
041614  002162
041616  001254  001116  001162        DT0063: .WORD   $TESTN,$ERRPC,$REG0,$REG3,$REG1,$REG4,$REG2
041624  001170  001164  001172
041632  001166
```

```
 8360                                   .SBTTL  DATA FORMAT FOR PRINT OUT
 8361
 8362  041634  000001                   DF000:  .WORD   1
 8363  041636     002     000                   .BYTE   2,0
 8364  041640  000006                   DF0004: .WORD   6
 8365  041642     000     000                   .BYTE   0,0
 8366  041644  043040                           .WORD   DH000A
 8367  041646     000     000                   .BYTE   0,0
 8368  041650  043056                           .WORD   DH000B
 8369  041652     002     000                   .BYTE   2,0
 8370  041654  043122                           .WORD   DH004
 8371  041656     000     000                   .BYTE   0,0
 8372  041660  043141                           .WORD   DH004A
 8373  041662     002     000                   .BYTE   2,0
 8374  041664  043231                           .WORD   DH004E
 8375  041666     003     000                   .BYTE   3,0
 8376  041670  000005                   DF0005: .WORD   5
 8377  041672     000     000                   .BYTE   0,0
 8378  041674  043040                           .WORD   DH000A
 8379  041676     000     000                   .BYTE   0,0
 8380  041700  043056                           .WORD   DH000B
 8381  041702     002     000                   .BYTE   2,0
 8382  041704  043122                           .WORD   DH004
 8383  041706     000     000                   .BYTE   0,0
 8384  041710  043157                           .WORD   DH004B
 8385  041712     002     000                   .BYTE   2,0
 8386  041714  000005                   DF0006: .WORD   5
 8387  041716     000     000                   .BYTE   0,0
 8388  041720  043040                           .WORD   DH000A
 8389  041722     000     000                   .BYTE   0,0
 8390  041724  043056                           .WORD   DH000B
 8391  041726     002     000                   .BYTE   2,0
 8392  041730  043122                           .WORD   DH004
 8393  041732     000     000                   .BYTE   0,0
 8394  041734  043175                           .WORD   DH004C
 8395  041736     002     000                   .BYTE   2,0
 8396  041740  000005                   DF0007: .WORD   5
 8397  041742     000     000                   .BYTE   0,0
 8398  041744  043040                           .WORD   DH000A
 8399  041746     000     000                   .BYTE   0,0
 8400  041750  043056                           .WORD   DH000B
 8401  041752     002     000                   .BYTE   2,0
 8402  041754  043122                           .WORD   DH004
 8403  041756     000     000                   .BYTE   0,0
 8404  041760  043213                           .WORD   DH004D
 8405  041762     002     000                   .BYTE   2,0
 8406  041764  000005                   DF0010: .WORD   5
 8407  041766     000     000                   .BYTE   0,0
 8408  041770  043040                           .WORD   DH000A
 8409  041772     000     000                   .BYTE   0,0
 8410  041774  043056                           .WORD   DH000B
 8411  041776     002     000                   .BYTE   2,0
 8412  042000  043325                           .WORD   DH010
 8413  042002     000     000                   .BYTE   0,0
 8414  042004  043371                           .WORD   DH010A
 8415  042006     005     000                   .BYTE   5,0
```

```
8416  042010  000006                  DF0011: .WORD   6
8417  042012     000      000                .BYTE   0,0
8418  042014  043040                         .WORD   DH000A
8419  042016     000      000                .BYTE   0,0
8420  042020  043056                         .WORD   DH000B
8421  042022     002      000                .BYTE   2,0
8422  042024  043325                         .WORD   DH10
8423  042026     000      000                .BYTE   0,0
8424  042030  043437                         .WORD   DH10B
8425  042032     005      000                .BYTE   5,0
8426  042034  043231                         .WORD   DH004E
8427  042036     003      000                .BYTE   3,0
8428  042040  000005                  DF0014: .WORD   5
8429  042042     000      000                .BYTE   0,0
8430  042044  043040                         .WORD   DH000A
8431  042046     000      000                .BYTE   0,0
8432  042050  043056                         .WORD   DH000B
8433  042052     002      000                .BYTE   2,0
8434  042054  043505                         .WORD   DH14
8435  042056     000      000                .BYTE   0,0
8436  042060  043544                         .WORD   DH14A
8437  042062     004      000                .BYTE   4,0
8438  042064  000005                  DF0017: .WORD   5
8439  042066     000      000                .BYTE   0,0
8440  042070  043040                         .WORD   DH000A
8441  042072     000      000                .BYTE   0,0
8442  042074  043056                         .WORD   DH000B
8443  042076     002      000                .BYTE   2,0
8444  042100  043601                         .WORD   DH0017
8445  042102     000      000                .BYTE   0,0
8446  042104  043665                         .WORD   DH17A
8447  042106     007      000                .BYTE   7,0
8448  042110  000005                  DF0021: .WORD   5
8449  042112     000      000                .BYTE   0,0
8450  042114  043040                         .WORD   DH000A
8451  042116     000      000                .BYTE   0,0
8452  042120  043056                         .WORD   DH000B
8453  042122     002      000                .BYTE   2,0
8454  042124  043753                         .WORD   DH0021
8455  042126     000      000                .BYTE   0,0
8456  042130  043777                         .WORD   DH021A
8457  042132     003      000                .BYTE   3,0
8458  042134  000004                  DF0026: .WORD   4
8459  042136     000      000                .BYTE   0,0
8460  042140  043040                         .WORD   DH000A
8461  042142     000      000                .BYTE   0,0
8462  042144  043056                         .WORD   DH000B
8463  042146     002      000                .BYTE   2,0
8464  042150  044053                         .WORD   DH0026
8465  042152     007      000                .BYTE   7,0
8466  042154  000003                  DF0027: .WORD   3
8467  042156     000      000                .BYTE   0,0
8468  042160  043056                         .WORD   DH000B
8469  042162     002      000                .BYTE   2,0
8470  042164  044053                         .WORD   DH0026
8471  042166     007      000                .BYTE   7,0
```

```
8472  042170  000004              DF0037: .WORD   4
8473  042172     000      000             .BYTE   0,0
8474  042174  043056                      .WORD   DH000B
8475  042176     002      000             .BYTE   2,0
8476  042200  044140                      .WORD   DH037A
8477  042202     000      000             .BYTE   0,0
8478  042204  044165                      .WORD   DH037B
8479  042206     003      000             .BYTE   3,0
8480  042210  000000              DF0040: .WORD   0
8481  042212     003      000             .BYTE   3,0
8482  042214  000004              DF0041: .WORD   4
8483  042216     000      000             .BYTE   0,0
8484  042220  043056                      .WORD   DH000B
8485  042222     002      000             .BYTE   2,0
8486  042224  044214                      .WORD   DH0041
8487  042226     000      000             .BYTE   0,0
8488  042230  044260                      .WORD   DH041A
8489  042232     005      000             .BYTE   5,0
8490  042234  000004              DF0042: .WORD   4
8491  042236     000      000             .BYTE   0,0
8492  042240  043056                      .WORD   DH000B
8493  042242     002      000             .BYTE   2,0
8494  042244  043753                      .WORD   DH0021
8495  042246     000      000             .BYTE   0,0
8496  042250  043777                      .WORD   DH021A
8497  042252     003      000             .BYTE   3,0
8498  042254  000004              DF0043: .WORD   4
8499  042256     000      000             .BYTE   0,0
8500  042260  043056                      .WORD   DH000B
8501  042262     002      000             .BYTE   2,0
8502  042264  043753                      .WORD   DH0021
8503  042266     000      000             .BYTE   0,0
8504  042270  044025                      .WORD   DH021B
8505  042272     003      000             .BYTE   3,0
8506  042274  000005              DF0045: .WORD   5
8507  042276     000      000             .BYTE   0,0
8508  042300  043040                      .WORD   DH000A
8509  042302     000      000             .BYTE   0,0
8510  042304  043056                      .WORD   DH000B
8511  042306     002      000             .BYTE   2,0
8512  042310  043122                      .WORD   DH004
8513  042312     000      000             .BYTE   0,0
8514  042314  043256                      .WORD   DH004F
8515  042316     002      000             .BYTE   2,0
8516  042320  000004              DF0046: .WORD   4
8517  042322     000      000             .BYTE   0,0
8518  042324  043056                      .WORD   DH000B
8519  042326     002      000             .BYTE   2,0
8520  042330  043122                      .WORD   DH004
8521  042332     000      000             .BYTE   0,0
8522  042334  043273                      .WORD   DH004G
8523  042336     002      000             .BYTE   2,0
8524  042340  000005              DF0050: .WORD   5
8525  042342     000      000             .BYTE   0,0
8526  042344  043040                     DH000A
8527  042346     000      000             .BYTE   0,0
```

```
 8528  042350  043056                        .WORD   DH000B
 8529  042352     002     000                .BYTE   2,0
 8530  042354  043122                        .WORD   DH004
 8531  042356     000     000                .BYTE   0,0
 8532  042360  043141                        .WORD   DH004A
 8533  042362     002     000                .BYTE   2,0
 8534  042364  000004             DF0060:    .WORD   4
 8535  042366     000     000                .BYTE   0,0
 8536  042370  043056                        .WORD   DH000B
 8537  042372     002     000                .BYTE   2,0
 8538  042374  044326                        .WORD   DH060A
 8539  042376     000     000                .BYTE   0,0
 8540  042400  044353                        .WORD   DH060B
 8541  042402     003     000                .BYTE   3,0
 8542  042404  000005             DF0061:    .WORD   5
 8543  042406     000     000                .BYTE   0,0
 8544  042410  043040                        .WORD   DH000A
 8545  042412     000     000                .BYTE   0,0
 8546  042414  043056                        .WORD   DH000B
 8547  042416     002     000                .BYTE   2,0
 8548  042420  043122                        .WORD   DH004
 8549  042422     000     000                .BYTE   0,0
 8550  042424  043273                        .WORD   DH004G
 8551  042426     002     000                .BYTE   2,0
 8552  042430  000005             DF0062:    .WORD   5
 8553  042432     000     000                .BYTE   0,0
 8554  042434  043040                        .WORD   DH000A
 8555  042436     000     000                .BYTE   0,0
 8556  042440  043056                        .WORD   DH000B
 8557  042442     002     000                .BYTE   2,0
 8558  042444  043122                        .WORD   DH004
 8559  042446     000     000                .BYTE   0,0
 8560  042450  043310                        .WORD   DH004H
 8561  042452     002     000                .BYTE   2,0
 8562  042454  000005             DF0063:    .WORD   5
 8563  042456     000     000                .BYTE   0,0
 8564  042460  043040                        .WORD   DH000A
 8565  042462     000     000                .BYTE   0,0
 8566  042464  043056                        .WORD   DH000B
 8567  042466     002     000                .BYTE   2,0
 8568  042470  044400                        .WORD   DH063A
 8569  042472     000     000                .BYTE   0,0
 8570  042474  044445                        .WORD   DH063B
 8571  042476     005     000                .BYTE   5,0
```

```
                                .SBTTL  ASCII MESSAGES

042500  005015  045522  030466  OPR001: .ASCIZ  <15><12>/RK611 VECTOR ADDRESS ( /
042506  020061  042526  052103
042514  051117  040440  042104
042522  042522  051523  024040
042530  000040
042532  024440  036440  000040  OPR002: .ASCIZ  / ) = /
042540  045522  030466  020061  OPR003: .ASCIZ  /RK611 VECTOR ADDRESS ( /
042546  042526  052103  051117
042554  040440  042104  042522
042562  051523  024040  000040
042570  045522  030466  020061  OPR004: .ASCIZ  /RK611 PRIORITY ( /
042576  051120  047511  044522
042604  054524  024040  000040
042612  005015  047062  020104  OPR006: .ASCIZ  <15><12>/2ND PASS RUN TIME APPROX 4 MINUTES/<15><12>
042620  040520  051523  051040
042626  047125  052040  046511
042634  020105  050101  051120
042642  054117  032040  046440
042650  047111  052125  051505
042656  005015     000
042661     015  025012  025052  OPR007: .ASCIZ  <15><12>/*****   PROGRAM HALTED   *****/<15><12>
042666  025052  020040  050040
042674  047522  051107  046501
042702  044040  046101  042524
042710  020104  020040  025052
042716  025052  006452  000012
042724  020040     000          SPACE2: .ASCIZ  /  /
042727     015  050012  047522  ABORT:  .ASCIZ  <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
042734  051107  046501  040440
042742  047502  052122  042105
042750  041040  041505  052501
042756  042523  042440  051122
042764  051117  052040  051110
042772  051505  047510  042114
043000  042440  041530  042505
043006  042504  006504  000012
043014  005015  042524  052123  TSTBY1: .ASCIZ  <15><12>/TEST /
043022  000040
043024  041040  050131  051501  TSTBY2: .ASCIZ  / BYPASSED/<15><12>
043032  042523  006504  000012
```

```
                                        .SBTTL  DATA HEADERS
                                        
043040  042524  052123  020040  DH000A: .ASCIZ  /TEST    ERROR/
043046  020040  051105  047522
043054  000122
043056  052516  020115  020040  DH000B: .ASCIZ  /NUM     PC/
043064  020040  041520    000
043071     124  051505  020124  DH000C: .ASCII  /TEST    TRAP/<15><12>
043076  020040  052040  040522
043104  006520     012
043107     116  046525  020040          .ASCIZ  /NUM     PC/
043114  020040  050040  000103
043122  054105  042520  052103  DH004:  .ASCIZ  /EXPECT  ACTUAL/
043130  020040  041501  052524
043136  046101     000
043141     122  041513  030523  DH004A: .ASCIZ  /RKCS1   RKCS1/
043146  020040  051040  041513
043154  030523     000
043157     122  046513  030522  DH004B: .ASCIZ  /RKMR1   RKMR1/
043164  020040  051040  046513
043172  030522     000
043175     122  046513  031122  DH004C: .ASCIZ  /RKMR2   RKMR2/
043202  020040  051040  046513
043210  031122     000
043213     122  046513  031522  DH004D: .ASCIZ  /RKMR3   RKMR3/
043220  020040  051040  046513
043226  031522     000
043231     122  041513  031123  DH004E: .ASCIZ  /RKCS2   RKER    RKDS/
043236  020040  051040  042513
043244  020122  020040  051040
043252  042113  000123
043256  045522  051105  020040  DH004F: .ASCIZ  /RKER    RKER/
043264  020040  045522  051105
043272     000
043273     122  041113  020101  DH004G: .ASCIZ  /RKBA    RKBA/
043300  020040  051040  041113
043306  000101
043310  045522  041527  020040  DH004H: .ASCIZ  /RKWC    RKWC/
043316  020040  045522  041527
043324     000
043325     105  050130  041505  DH010:  .ASCIZ  /EXPECT  ACTUAL  PRESENT PREVOUS BIT/
043332  020124  040440  052103
043340  040525  020114  050040
043346  042522  042523  052116
043354  050040  042522  047526
043362  051525  041040  052111
043370     000
043371     122  046513  030522  DH010A: .ASCIZ  /RKMR1   RKMR1   BIT     BIT     COUNT/
043376  020040  051040  046513
043404  030522  020040  041040
043412  0521.1  020040  020040
043420  041040  052111  020040
043426  020040  041440  052517
043434  052116     000
043437     122  041513  030523  DH010B: .ASCIZ  /RKCS1   RKCS1   BIT     BIT     COUNT/
043444  020040  051040  041513
```

```
8670  043452  030523  020040  041040
8671  043460  052111  020040  020040
8672  043466  041040  052111  020040
8673  043474  020040  041440  052517
8674  043502  052116     000
8675  043505     105  050130  041505  DH014:  .ASCIZ  /EXPECT  ACTUAL  EXPECT  ACTUAL/
8676  043512  020124  040440  052103
8677  043520  040525  020114  042440
8678  043526  050130  041505  020124
8679  043534  040440  052103  040525
8680  043542  000114
8681  043544  045522  041504  046131  DH014A: .ASCIZ  /RKDCYL  RKDCYL  RKDA    RKDA/
8682  043552  020040  045522  041504
8683  043560  046131  020040  045522
8684  043566  040504  020040  020040
8685  043574  045522  040504     000
8686  043601     105  050130  041505  DH0017: .ASCIZ  /EXPECT  ACTUAL  PRESENT PRESENT PRESENT PRESENT BIT/
8687  043606  020124  040440  052103
8688  043614  040525  020114  050040
8689  043622  042522  042523  052116
8690  043630  050040  042522  042523
8691  043636  052116  050040  042522
8692  043644  042523  052116  050040
8693  043652  042522  042523  052116
8694  043660  041040  052111     000
8695  043665     122  046513  030522  DH017A: .ASCIZ  /RKMR1   RKMR1   BIT+1   BIT     BIT-1   BIT-2   COUNT/
8696  043672  020040  051040  046513
8697  043700  030522  020040  041040
8698  043706  052111  030453  020040
8699  043714  041040  052111  020040
8700  043722  020040  041040  052111
8701  043730  030455  020040  041040
8702  043736  052111  031055  020040
8703  043744  041440  052517  052116
8704  043752     000
8705  043753     105  050130  041505  DH0021: .ASCIZ  /EXPECT  ACTUAL  BIT/
8706  043760  020124  040440  052103
8707  043766  040525  020114  041040
8708  043774  052111     000
8709  043777     122  042513  050103  DH021A: .ASCIZ  /RKECPT  RKECPT  COUNT/
8710  044004  020124  051040  042513
8711  044012  050103  020124  041440
8712  044020  052517  052116     000
8713  044025     122  042513  050103  DH021B: .ASCIZ  /RKECPS  RKECPS  COUNT/
8714  044032  020123  051040  042513
8715  044040  050103  020123  041440
8716  044046  052517  052116     000
8717  044053     122  041513  030523  DH0026: .ASCIZ  /RKCS1   RKCS2   RKER            RKWC    RKDCYL  RKDA/
8718  044060  020040  051040  041513
8719  044066  031123  020040  051040
8720  044074  042513  020122  020040
8721  044102  051040  042113  020123
8722  044110  020040  051040  053513
8723  044116  020103  020040  051040
8724  044124  042113  054503  020114
8725  044132  051040  042113  000101
```

```
8726  044140  047507  042117  020040  DH037A: .ASCIZ  /GOOD    BAD     WORD/
8727  044146  020040  040502  020104
8728  044154  020040  020040  047527
8729  044162  042122     000
8730  044165     127  051117  020104  DH037B: .ASCIZ  /WORD    WORD    NUMBER/
8731  044172  020040  053440  051117
8732  044200  020104  020040  047040
8733  044206  046525  042502  000122
8734  044214  054105  042520  052103  DH0041: .ASCIZ  /EXPECT  ACTUAL  EXPECT  ACTUAL  BIT/
8735  044222  020040  041501  052524
8736  044230  046101  020040  054105
8737  044236  042520  052103  020040
8738  044244  041501  052524  046101
8739  044252  020040  044502  000124
8740  044260  045522  041505  052120  DH041A: .ASCIZ  /RKECPT  RKECPT  RKECPS  RKECPS  COUNT/
8741  044266  020040  045522  041505
8742  044274  052120  020040  045522
8743  044302  041505  051520  020040
8744  044310  045522  041505  051520
8745  044316  020040  047503  047125
8746  044324  000124
8747  044326  045522  051503  020062  DH060A: .ASCIZ  /RKCS2   BUFFER  WORD/
8748  044334  020040  052502  043106
8749  044342  051105  020040  047527
8750  044350  042122     000
8751  044353     040  020040  020040  DH060B: .ASCIZ  /        WORD    READ/
8752  044360  020040  053440  051117
8753  044366  020104  020040  051040
8754  044374  040505  000104
8755  044400  045522  051503  020062  DH063A: .ASCIZ  /RKCS2   BA17-16 BUFFER  BA17-16 WORD/
8756  044406  020040  040502  033461
8757  044414  030455  020066  052502
8758  044422  043106  051105  020040
8759  044430  040502  033461  030455
8760  044436  020066  047527  042122
8761  044444     000
8762  044445     040  020040  020040  DH063B: .ASCIZ  /        BUF WD  WORD    WD READ READ/
8763  044452  020040  041040  043125
8764  044460  053440  020104  053440
8765  044466  051117  020104  020040
8766  044474  053440  020104  042522
8767  044502  042101  051040  040505
8768  044510  000104
```

```
8769                                    .SBTTL  ERROR MESSAGES
8770
8771  044512  047125  054105  042520  EM000:  .ASCIZ  /UNEXPECTED MEMORY PARITY ENABLE TRAP/
8772  044520  052103  042105  046440
8773  044526  046505  051117  020131
8774  044534  040520  044522  054524
8775  044542  042440  040516  046102
8776  044550  020105  051124  050101
8777  044556     000
8778  044557     105  051122  051117  EM5004: .ASCIZ  /ERROR AFTER ATTEMPTING IMPLIED SEEK/
8779  044564  040440  052106  051105
8780  044572  040440  052124  046505
8781  044600  052120  047111  020107
8782  044606  046511  046120  042511
8783  044614  020104  042523  045505
8784  044622     000
8785  044623     122  041513  030523  E5004A: .ASCIZ  /RKCS1 IN ERROR/
8786  044630  044440  020116  051105
8787  044636  047522  000122
8788  044642  045522  051115  020061  E5004B: .ASCIZ  /RKMR1 IN ERROR/
8789  044650  047111  042440  051122
8790  044656  051117     000
8791  044661     122  046513  031122  E5004C: .ASCIZ  /RKMR2 IN ERROR/
8792  044666  044440  020116  051105
8793  044674  047522  000122
8794  044700  045522  051115  020063  E5004D: .ASCIZ  /RKMR3 IN ERROR/
8795  044706  047111  042440  051122
8796  044714  051117     000
8797  044717     122  042513  020122  E5004E: .ASCIZ  /RKER IN ERROR/
8798  044724  047111  042440  051122
8799  044732  051117     000
8800  044735     122  041113  020101  E5004F: .ASCIZ  /RKBA IN ERROR/
8801  044742  047111  042440  051122
8802  044750  051117     000
8803  044753     122  053513  020103  E5004G: .ASCIZ  /RKWC IN ERROR/
8804  044760  047111  042440  051122
8805  044766  051117     000
8806  044771     105  051122  051117  EM5010: .ASCIZ  /ERROR ATTEMPTING TO READ HEADER SYNC (PREAMBLE)/
8807  044776  040440  052124  046505
8808  045004  052120  047111  020107
8809  045012  047524  051040  040505
8810  045020  020104  042510  042101
8811  045026  051105  051440  047131
8812  045034  020103  050050  042522
8813  045042  046501  046102  024505
8814  045050     000
8815  045051     105  051122  051117  EM5011: .ASCIZ  &ERROR ATTEMPTING TO DO HEADER READ/COMPARE&
8816  045056  040440  052124  046505
8817  045064  052120  047111  020107
8818  045072  047524  042040  020117
8819  045100  042510  042101  051105
8820  045106  051040  040505  027504
8821  045114  047503  050115  051101
8822  045122  000105
8823  045124  045522  041504  046131  E5011A: .ASCIZ  /RKDCYL OR RKDA IN ERROR/
8824  045132  047440  020122  045522
```

```
8825   045140  040504  044440  020116
8826   045146  051105  047522  000122
8827   045154  051105  047522  020122  EM5015: .ASCIZ  /ERROR ATTEMFTING TO READ GAP/
8828   045162  052101  042524  050115
8829   045170  044524  043516  052040
8830   045176  020117  042522  042101
8831   045204  043440  050101     000
8832   045211     105  051122  051117  EM5020: .ASCIZ  /ERROR ATTEMPTING TO WRITE DATA SYNC (PREAMBLE)/
8833   045216  040440  052124  046505
8834   045224  052120  047111  020107
8835   045232  047524  053440  044522
8836   045240  042524  042040  052101
8837   045246  020101  054523  041516
8838   045254  024040  051120  040505
8839   045262  041115  042514  000051
8840   045270  051105  047522  020122  EM5021: .ASCIZ  /ERROR ATTEMPTING TO WRITE DATA OR ECC/
8841   045276  052101  042524  050115
8842   045304  044524  043516  052040
8843   045312  020117  051127  052111
8844   045320  020105  040504  040524
8845   045326  047440  020122  041505
8846   045334  000103
8847   045336  051105  047522  020122  E5021A: .ASCIZ  /ERROR IN ECC GENERATION/
8848   045344  047111  042440  041503
8849   045352  043440  047105  051105
8850   045360  052101  047511  000116
8851   045366  051105  047522  020122  EM5023: .ASCIZ  /ERROR BETWEEN SECTORS ATTEMPTING MULTI-SECTOR WRITE/
8852   045374  042502  053524  042505
8853   045402  020116  042523  052103
8854   045410  051117  020123  052101
8855   045416  042524  050115  044524
8856   045424  043516  046440  046125
8857   045432  044524  051455  041505
8858   045440  047524  020122  051127
8859   045446  052111  000105
8860   045452  051115  020061  047111  EMW1:   .ASCIZ  /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
8861   045460  047503  051122  041505
8862   045466  020124  047117  030440
8863   045474  052123  052440  053520
8864   045502  051101  020104  051124
8865   045510  047101  044523  044524
8866   045516  047117  047440  020106
8867   045524  040515  047111  020124
8868   045532  046103  041517  000113
8869   045540  051115  020061  047111  EMW2:   .ASCIZ  /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
8870   045546  047503  051122  041505
8871   045554  020124  047117  030440
8872   045562  052123  042040  053517
8873   045570  053516  051101  020104
8874   045576  051124  047101  044523
8875   045604  044524  047117  047440
8876   045612  020106  040515  047111
8877   045620  020124  046103  041517
8878   045626  000113
8879   045630  051115  020061  047111  EMW3:   .ASCIZ  /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/
8880   045636  047503  051122  041505
```

```
8881  045644  020124  047117  031040
8882  045652  042116  052440  053520
8883  045660  051101  020104  051124
8884  045666  047101  044523  044524
8885  045674  047117  047440  020106
8886  045702  040515  047111  020124
8887  045710  046103  041517  000113
8888  045716  051115  020061  047111  EMW4:   .ASCIZ  /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/
8889  045724  047503  051122  041505
8890  045732  020124  047117  031040
8891  045740  042116  042040  053517
8892  045746  053516  051101  020104
8893  045754  051124  047101  044523
8894  045762  044524  047117  047440
8895  045770  020106  040515  047111
8896  045776  020124  046103  041517
8897  046004  000113
8898  046006  054105  042520  052103  EM5026: .ASCIZ  /EXPECTED CYLINDER OVERFLOW ERR DID NOT SET/
8899  046014  042105  041440  046131
8900  046022  047111  042504  020122
8901  046030  053117  051105  046106
8902  046036  053517  042440  051122
8903  046044  042040  042111  047040
8904  046052  052117  051440  052105
8905  046060     000
8906  046061     127  044522  042524  E5026A: .ASCIZ  /WRITE TO CYL 632, TRACK 2, SECTOR 25, WORD COUNT 401/
8907  046066  052040  020117  054503
8908  046074  020114  031466  026062
8909  046102  052040  040522  045503
8910  046110  031040  020054  042523
8911  046116  052103  051117  031040
8912  046124  026065  053440  051117
8913  046132  020104  047503  047125
8914  046140  020124  030064  000061
8915  046146  047125  054105  042520  EM5027: .ASCIZ  /UNEXPECTED ERROR WHILE TRYING TO FORCE CYL OVERFLOW/
8916  046154  052103  042105  042440
8917  046162  051122  051117  053440
8918  046170  044510  042514  052040
8919  046176  054522  047111  020107
8920  046204  047524  043040  051117
8921  046212  042503  041440  046131
8922  046220  047440  042526  043122
8923  046226  047514  000127
8924  046232  051127  052111  020105  E5026B: .ASCIZ  /WRITE TO CYL 1452, TRACK 2, SECTOR 25, WORD COUNT 401/
8925  046240  047524  041440  046131
8926  046246  030440  032464  026062
8927  046254  052040  040522  045503
8928  046262  031040  020054  042523
8929  046270  052103  051117  031040
8930  046276  026065  053440  051117
8931  046304  020104  047503  047125
8932  046312  020124  030064  000061
8933  046320  047125  054105  042520  EM5030: .ASCIZ  /UNEXPECTED CYLINDER OVERFLOW WRITING LAST PHYSICAL ADDRESS/
8934  046326  052103  042105  041440
8935  046334  046131  047111  042504
8936  046342  020122  053117  051105
```

```
 8937  046350  046106  053517  053440
 8938  046356  044522  044524  043516
 8939  046364  046040  051501  020124
 8940  046372  044120  051531  041511
 8941  046400  046101  040440  042104
 8942  046406  042522  051523     000
 8943  046413     125  042516  050130  EM5031: .ASCIZ  /UNEXPECTED ERROR WHILE WRITING LAST PHYSICAL ADDRESS/
 8944  046420  041505  042524  020104
 8945  046426  051105  047522  020122
 8946  046434  044127  046111  020105
 8947  046442  051127  052111  047111
 8948  046450  020107  040514  052123
 8949  046456  050040  054510  044523
 8950  046464  040503  020114  042101
 8951  046472  051104  051505  000123
 8952  046500  051105  047522  020122  EM5032: .ASCIZ  /ERROR ATTEMPTING TO READ DATA SYNC (PREAMBLE)/
 8953  046506  052101  042524  050115
 8954  046514  044524  043516  052040
 8955  046522  020117  042522  042101
 8956  046530  042040  052101  020101
 8957  046536  054523  041516  024040
 8958  046544  051120  040505  041115
 8959  046552  042514  000051
 8960  046556  051105  047522  020122  EM5034: .ASCIZ  /ERROR ATTEMPTING TO READ DATA OR ECC/
 8961  046564  052101  042524  050115
 8962  046572  044524  043516  052040
 8963  046600  020117  042522  042101
 8964  046606  042040  052101  020101
 8965  046614  051117  042440  041503
 8966  046622     000
 8967  046623     105  041503  042440  EM5035: .ASCIZ  /ECC ERROR IN READ DATA OPERATION/
 8968  046630  051122  051117  044440
 8969  046636  020116  042522  042101
 8970  046644  042040  052101  020101
 8971  046652  050117  051105  052101
 8972  046660  047511  000116
 8973  046664  040504  040524  041440  EM5037: .ASCIZ  /DATA COMPARE ERROR AFTER INPUT NPR TRANSFER/
 8974  046672  046517  040520  042522
 8975  046700  042440  051122  051117
 8976  046706  040440  052106  051105
 8977  046714  044440  050116  052125
 8978  046722  047040  051120  052040
 8979  046730  040522  051516  042506
 8980  046736  000122
 8981  046740  041505  020103  042522  EM5041: .ASCIZ  /ECC REGISTER NOT CORRECT AFTER ECC READ IN READ DATA/
 8982  046746  044507  052123  051105
 8983  046754  047040  052117  041440
 8984  046762  051117  042522  052103
 8985  046770  040440  052106  051105
 8986  046776  042440  041503  051040
 8987  047004  040505  020104  047111
 8988  047012  051040  040505  020104
 8989  047020  040504  040524     000
 8990  047025     105  051122  051117  EM5042: .ASCIZ  /ERROR IN ECC PATTERN CALCULATION AFTER ECC ERROR/
 8991  047032  044440  020116  041505
 8992  047040  020103  040520  052124
```

```
 8993  047046  051105  020116  040503
 8994  047054  041514  046125  052101
 8995  047062  047511  020116  043101
 8996  047070  042524  020122  041505
 8997  047076  020103  051105  047522
 8998  047104  000122
 8999  047106  051105  047522  020122  EM5043: .ASCIZ  /ERROR IN ECC POSITION COUNTING AFTER ECC ERROR/
 9000  047114  047111  042440  041503
 9001  047122  050040  051517  052111
 9002  047130  047511  020116  047503
 9003  047136  047125  044524  043516
 9004  047144  040440  052106  051105
 9005  047152  042440  041503  042440
 9006  047160  051122  051117     000
 9007  047165     105  051122  051117  EM5044: .ASCIZ  /ERROR AFTER PROCESSING DATA CHECK ERROR/
 9008  047172  040440  052106  051105
 9009  047200  050040  047522  042503
 9010  047206  051523  047111  020107
 9011  047214  040504  040524  041440
 9012  047222  042510  045503  042440
 9013  047230  051122  051117     000
 9014  047235     124  040522  051516  EM5046: .ASCIZ  /TRANSFER LENGTH ERROR ATTEMPTING PARTIAL SECTOR OPERATION/
 9015  047242  042506  020122  042514
 9016  047250  043516  044124  042440
 9017  047256  051122  051117  040440
 9018  047264  052124  046505  052120
 9019  047272  047111  020107  040520
 9020  047300  052122  040511  020114
 9021  047306  042523  052103  051117
 9022  047314  047440  042520  040522
 9023  047322  044524  047117     000
 9024  047327     102  042101  051440  EM5047: .ASCIZ  /BAD SECTOR ERROR DID NOT PREVENT DESIRED ADDRESS INCREMENT/
 9025  047334  041505  047524  020122
 9026  047342  051105  047522  020122
 9027  047350  044504  020104  047516
 9028  047356  020124  051120  053105
 9029  047364  047105  020124  042504
 9030  047372  044523  042522  020104
 9031  047400  042101  051104  051505
 9032  047406  020123  047111  051103
 9033  047414  046505  047105  000124
 9034  047422  040502  020104  042523  EM5050: .ASCIZ  /BAD SECTOR ERROR DID NOT CAUSE CERR TO SET/
 9035  047430  052103  051117  042440
 9036  047436  051122  051117  042040
 9037  047444  042111  047040  052117
 9038  047452  041440  052501  042523
 9039  047460  041440  051105  020122
 9040  047466  047524  051440  052105
 9041  047474     000
 9042  047475     102  042101  051440  EM5051: .ASCIZ  /BAD SECTOR ERROR FAILED TO SET WHEN EXPECTED/
 9043  047502  041505  047524  020122
 9044  047510  051105  047522  020122
 9045  047516  040506  046111  042105
 9046  047524  052040  020117  042523
 9047  047532  020124  044127  047105
 9048  047540  042440  050130  041505
```

```
9049  047546  042524  000104
9050  047552  040504  040524  052040  EM5052: .ASCII  /DATA TRANSFER DID NOT CONTINUE TO END OF SECTOR/
9051  047560  040522  051516  042506
9052  047566  020122  044504  020104
9053  047574  047516  020124  047503
9054  047602  052116  047111  042525
9055  047610  052040  020117  047105
9056  047616  020104  043117  051440
9057  047624  041505  047524     122
9058  047631     015  053412  042510          .ASCIZ  <15><12>/WHEN BAD SECTOR ERROR OCCURRED/
9059  047636  020116  040502  020104
9060  047644  042523  052103  051117
9061  047652  042440  051122  051117
9062  047660  047440  041503  051125
9063  047666  042522  000104
9064  047672  051105  047522  020122  EM5053: .ASCIZ  /ERROR ATTEMPTING WRITE CHECK OR ECC READ AFTER WRT CHK/
9065  047700  052101  042524  050115
9066  047706  044524  043516  053440
9067  047714  044522  042524  041440
9068  047722  042510  045503  047440
9069  047730  020122  041505  020103
9070  047736  042522  042101  040440
9071  047744  052106  051105  053440
9072  047752  052122  041440  045510
9073  047760     000
9074  047761     105  041503  042440  EM5054: .ASCIZ  /ECC ERROR IN WRITE CHECK OPERATION/
9075  047766  051122  051117  044440
9076  047774  020116  051127  052111
9077  050002  020105  044103  041505
9078  050010  020113  050117  051105
9079  050016  052101  047511  000116
9080  050024  051127  052111  020105  EM5060: .ASCIZ  /WRITE CHECK ERROR FAILED TO SET IN 16 BIT MODE/
9081  050032  044103  041505  020113
9082  050040  051105  047522  020122
9083  050046  040506  046111  042105
9084  050054  052040  020117  042523
9085  050062  020124  047111  030440
9086  050070  020066  044502  020124
9087  050076  047515  042504     000
9088  050103     124  040522  051516  EM5061: .ASCIZ  /TRANSFER LENGTH ERROR WITH WRITE CHECK ERROR/
9089  050110  042506  020122  042514
9090  050116  043516  044124  042440
9091  050124  051122  051117  053440
9092  050132  052111  020110  051127
9093  050140  052111  020105  044103
9094  050146  041505  020113  051105
9095  050154  047522  000122
9096  050160  051127  052111  020105  EM5063: .ASCIZ  /WRITE CHECK ERROR FAILED TO SET IN 18 BIT MODE/
9097  050166  044103  041505  020113
9098  050174  051105  047522  020122
9099  050202  040506  046111  042105
9100  050210  052040  020117  042523
9101  050216  020124  047111  030440
9102  050224  020070  044502  020124
9103  050232  047515  042504     000
9104
```

```
                        050240                          .EVEN
                                                .SBTTL  DATA PATTERNS AND BUFFERS
                                                ;*      PATTERN 1 - ALL ZEROS
                                                ;*      PATTERN 2 - ROTATING CELL PULSE PRECOMP

                050240                          PAT2:
                050240  121105                          121105
                050242  150442                          150442
                050244  064221                          064221
                050246  132110                          132110
                050250  055044                          055044
                050252  026422                          026422
                050254  013211                          013211
                050256  105504                          105504
                050260  042642                          042642
                050262  021321                          021321
                050264  110550                          110550
                050266  044264                          044264
                050270  022132                          022132
                050272  011055                          011055
                050274  104426                          104426
                050276  042213                          042213

                                                ;*      PATTERN 3 - MAX PRECOMP PHASE MIX

                050300                          PAT3:
                050300  133333                          133333
                050302  066666                          066666
                050304  155555                          155555
                050306  155555                          155555
                050310  133333                          133333
                050312  066666                          066666
                050314  066666                          066666
                050316  155555                          155555
                050320  155555                          155555
                050322  133333                          133333
                050324  133333                          133333
                050326  133333                          133333
                050330  133333                          133333
                050332  133333                          133333
                050334  133333                          133333
                050336  133333                          133333

                                                ;*      PATTERN 4 - HI-LO FREQ. MIX

                050340                          PAT4:
                050340  177777                          177777
                050342  177777                          177777
                050344  177777                          177777
                050346  052525                          052525
                050350  052525                          052525
                050352  052525                          052525
                050354  177777                          177777
                050356  177777                          177777
                050360  052525                          052525
                050362  052525                          052525
```

```
9161  050364  177777                    177777
9162  050366  052525                    052525
9163  050370  177252                    177252
9164  050372  177252                    177252
9165  050374  172765                    172765
9166  050376  172765                    172765
9167
9168                          ;*      PATTERN 5 - ALTERNATING 1'S AND 0'S
9169
9170                          ;*      PATTERN 6 - COMPOSIT ROTATING
9171
9172  050400                  PAT6:
9173  050400  072307                    072307
9174  050402  135143                    135143
9175  050404  156461                    156461
9176  050406  167230                    167230
9177  050410  073514                    073514
9178  050412  035646                    035646
9179  050414  016723                    016723
9180  050416  107351                    107351
9181  050420  143564                    143564
9182  050422  061672                    061672
9183  050424  030735                    030735
9184  050426  114356                    114356
9185  050430  046167                    046167
9186  050432  123073                    123073
9187  050434  151453                    151453
9188  050436  164616                    164616
9189
9190                          ;*      PATTERN 7 - ALL ONES
9191
9192  050440  000402          IBUFF:  .BLKW   402
9193  051444  000402          OBUFF:  .BLKW   402
9194          000001          .END
```

```
ABASE = 177440          1098#   1351    1392
ABORT   042727          7633    8601#
ACDW1 = 000000          1351
ACDW2 = 000000          1351
ACLO  = 000010          1193#
ACPUOP= 000000          1351    1366
ADDW0 = 000000          1351
ADDW1 = 000000          1351
ADDW10= 000000          1351
ADDW11= 000000          1351
ADDW12= 000000          1351
ADDW13= 000000          1351
ADDW14= 000000          1351
ADDW15= 000000          1351
ADDW2 = 000000          1351
ADDW3 = 000000          1351
ADDW4 = 000000          1351
ADDW5 = 000000          1351
ADDW6 = 000000          1351
ADDW7 = 000000          1351
ADDW8 = 000000          1351
ADDW9 = 000000          1351
ADEVCT= 000000          1351    1357
ADEVM = 000000          1351    1393
AENV  = 000000          1351    1362
AENVM = 000000          1351    1363
AFATAL= 000000          1351    1354
AMADR1= 000000          1351    1379
AMADR2= 000000          1351    1383
AMADR3= 000000          1351    1386
AMADR4= 000000          1351    1389
AMAMS1= 000000          1351    1373
AMAMS2= 000000          1351    1381
AMAMS3= 000000          1351    1384
AMAMS4= 000000          1351    1387
AMSGAD= 000000          1351    1359
AMSGLG= 000000          1351    1360
AMSGTY= 000000          1351    1353
AMTYP1= 000000          1351    1374
AMTYP2= 000000          1351    1382
AMTYP3= 000000          1351    1385
AMTYP4= 000000          1351    1388
APASS = 000000          1351    1356
APRIOR= 000005          1097#   1351
APTCSU= 000040          7508#   7689
APTENV= 000001          7464    7506#   7540    7682
APTSIZ= 000200          1850    7505#
APTSPO= 000100          7466    7507#   7684
ASWREG= 000000          1351    1364
ATESTN= 000000          1351    1355
AUNIT = 000000          1351    1358
AUSWR = 000000          1351    1365
AVECT1= 120210          1096#   1351    1390
AVECT2= 000000          1351    1391
BADPAR  002334          1784#   7231*
BAI   = 000020          1156#
```

```
BA16   = 000400          1141#
BA17   = 001000          1142#
BITCNT  002326           1781#  3372*  3389*  3502*  3519*  3635*  3652*  3766*  3783*  3897*  3914*  4028*  4045*
                         4160*  4177*  4483*  4500*  4614*  4631*  5076*  5093*  5900*  5925*  5938*  5953*  5996*
                         6055*  6188*  6193*  6195*  6208*  6232*  6242*  6253*  6290*  6322*  6365*  6391*  6466*
                         6471*  6473*  6486*  6510*  6521*  6533*  6545*  6632*  6661   6669*  6688   6714   6722*
                         6723   6726   6739   6744   8327   8330   8336   8339   8346   8349
BIT0   = 000001          1042#  1138   1171   1190   3728   3858   3989   4575   5351   5448   6027   6032   6304
                         6353   6379   6646   6702   6772   7210
BIT00  = 000001          1032#  1042
BIT01  = 000002          1031#  1041
BIT02  = 000004          1030#  1040
BIT03  = 000010          1029#  1039
BIT04  = 000020          1028#  1038
BIT05  = 000040          1027#  1037
BIT06  = 000100          1026#  1036
BIT07  = 000200          1025#  1035
BIT08  = 000400          1024#  1034   7364
BIT09  = 001000          1023#  1033   7382   7551
BIT1   = 000002          1041#  1172   3859   4576   5802   6843
BIT10  = 002000          1022#  1143   1162   1181   1214   1229   6095   6867   6869   7528
BIT11  = 004000          1021#  1144   1163   1182   1199   1215   1230   3725   3855   3986   4572   7389
BIT12  = 010000          1020#  1146   1164   1183   1216   3726   3856   3987   3990   4573   6106
BIT13  = 020000          1019#  1147   1165   1184   1200   1217   6863   6865   7535
BIT14  = 040000          1018#  1148   1166   1185   1201   1218   3726   3856   3987   4573   6006   6343   6679
                         7350
BIT15  = 100000          1017#  1149   1150   1167   1186   1202   1219   4837   6006   6076   7211
BIT2   = 000004          1040#  1173   1192
BIT3   = 000010          1039#  1155   1174   1193
BIT4   = 000020          1038#  1156   1175   1194   1208   1223   6863   6865
BIT5   = 000040          1037#  1157   1176   1195   1209   1224
BIT6   = 000100          1036#  1139   1158   1177   1196   1210   1225   3329   3331   3333   3459   3461   3463
                         3592   3594   3596   3726   3856   3987   4117   4119   4121   4440   4442   4444   4573
                         5032   5034   5036   5652   5654   5656   6100
BIT7   = 000200          1035#  1140   1159   1178   1197   1211   1226
BIT8   = 000400          1034#  1141   1160   1179   1198   1212   1227   6104   6867   6869
BIT9   = 001000          1033#  1142   1161   1180   1213   1228   2023   2141   2258   2375   2492   2609   2729
                         3725   3855   3986   4572   5830   5842   5851   5859   5867   5872   5915   5962   6010
                         6041   6097   6152   6214   6263   6417   6427   6492   6555   6804   6821
BLDDAT  033612           1969   2087   2204   2321   2438   2555   2675   2790   2873   2957   3040   3127   3234
                         3321   3451   3584   3717   3847   3978   4109   4240   4345   4432   4564   4696   4791
                         4925   5018   5154   5245   5638   7068#
BPTVEC= 000014           1049#
BSE    = 000200          1178#  4859   4861
BSEDES  002352           1791#  1959*  2077*  2194*  2311*  2428*  2545*  2662*  2779*  2862*  2947*  3029*  3112*
                         3219*  3307*  3437*  3570*  3702*  3833*  3964*  4095*  4225*  4330*  4418*  4550*  4681*
                         4778*  4837*  4853*  4911*  5006*  5141*  5232*  5337*  5434*  5527*  5627*  6016   6018
                         6019
CCLR   = 100000          1150#  1957   2075   2192   2309   2426   2543   2660   2780   2863   2945   3030   3113
                         3220   3308   3438   3571   3704   3834   3965   4096   4226   4331   4419   4551   4682
                         4776   4935   5039   5164   5255   5356   5453   5541   5659
CDT    = 002000          1143#  2436   2553   2955   3038   6093   6131
CERR   = 100000          1149#  2833   2919   3000   3086   3409   3539   3672   3803   3934   4065   4197   4520
                         4651   4881   5113
CFMT   = 010000          1146#  2021   2139   2256   2373   2490   2607   2673   2727   4236   4341   4428   4560
                         5150   5535   5808   5828   6008   6090   6123   6328   6339   6636   6675
```

```
CKSWR = 104407          7349    7524    7550    8308#
CLEAR = 000005          1123#
CLRIBF  033566          3125    3232    3319    3449    3582    3715    3845    3976    4107    4238    4343    4430    4562
                        4694    4789    4923    5016    5152    5243    5636    7057#
COE   = 001000          1180#   2836    2916    3002    3083
CR    = 000015           957#   7728    7738
CRLF  = 000200           958#   1875    7699    7738
CTO   = 004000          1144#
CTRHLT  036364          7645#   7924    8007
DCK   = 100000          1186#   3417    3547    3680    3811    3942    4073    4205    4528    4659    5121
DDISP = 177570           964#   1309    1838
DDT   = 000400          1198#
DF000   041634          1415    8362#
DF0004  041640          1430    1532    1622    1682    1688    8364#
DF0005  041670          1436    8376#
DF0006  041714          1442    1520    8386#
DF0007  041740          1448    1526    8396#
DF0010  041764          1454    1466    1484    1562    1574    1664    8406#
DF0011  042010          1460    1472    1490    1502    1514    1568    1586    1676    8416#
DF0014  042040          1478    1640    8428#
DF0017  042064          1496    8438#
DF0021  042110          1508    1580    1670    8448#
DF0026  042134          1538    1544    1550    1556    8458#
DF0027  042154          8466#
DF0037  042170          1592    8472#
DF0040  042210          1598    8480#
DF0041  042214          1604    8482#
DF0042  042234          1610    8490#
DF0043  042254          1616    8498#
DF0045  042274          1628    1652    8506#
DF0046  042320          1634    1658    8516#
DF0050  042340          1646    8524#
DF0060  042364          1694    8534#
DF0061  042404          1700    8542#
DF0062  042430          1706    8552#
DF0063  042454          1712    8562#
DHDCMP  026042          1986    2052    2104    2170    2221    2287    2338    2404    2455    2521    2572    2638    2692
                        2758    2807    2890    2974    3057    3144    3251    3349    3479    3612    3743    3874    4005
                        4137    4257    4362    4460    4591    4713    4808    4845    4949    5053    5178    5269    5370
                        5467    5560    5673    5991#
DH000A  043040          1542    1590    1602    1608    1614    1632    1656    1692    8366    8378    8388    8398    8408
                        8418    8430    8440    8450    8460    8508    8526    8544    8554    8564    8616#
DH000B  043056          8368    8380    8390    8400    8410    8420    8432    8442    8452    8462    8468    8474    8484
                        8492    8500    8510    8518    8528    8536    8546    8556    8566    8619#
DH000C  043071          1413    8621#
DH0017  043601          8444    8686#
DH0021  043753          8454    8494    8502    8705#
DH0026  044053          8464    8470    8717#
DH004   043122          8370    8382    8392    8402    8512    8520    8530    8548    8558    8626#
DH004A  043141          8372    8532    8629#
DH004B  043157          8384    8632#
DH004C  043175          8394    8635#
DH004D  043213          8404    8638#
DH004E  043231          8374    8426    8641#
DH004F  043256          8514    8645#
DH004G  043273          8522    8550    8648#
```

```
DH004H  043310          8560    8651#
DH0041  044214          8486    8734#
DH010   043325          8412    8422    8654#
DH010A  043371          8414    8661#
DH010B  043437          8424    8668#
DH014   043505          8434    8675#
DH014A  043544          8436    8681#
DH017A  043665          8446    8695#
DH021A  043777          8456    8496    8709#
DH021B  044025          8504    8713#
DH037A  044140          8476    8726#
DH037B  044165          8478    8730#
DH041A  044260          8488    8740#
DH060A  044326          8538    8747#
DH060B  044353          8540    8751#
DH063A  044400          8568    8755#
DH063B  044445          8570    8762#
DI    = 040000          1148#
DISEEK  025100          1974    2092    2209    2326    2443    2560    2680    2795    2878    2962    3045    3132    3239
                        3337    3467    3600    3731    3862    3993    4125    4245    4350    4448    4579    4701    4796
                        4937    5041    5166    5257    5358    5455    5548    5661    5799#
DISPLA  001142          1309#   1838*   1846*   7404*   7527*
DISPRE  000174          1238#   1846
DLT   = 100000          1167#
DMD   = 000040          1209#   2005    2006    2123    2124    2240    2241    2357    2358    2474    2475    2591    2592
                        2711    2712    2827    2828    2910    2911    2994    2995    3077    3078    3276    3277    4387
                        4388    4744    4745    4832    4833    4974    4975    5203    5204    5300    5301    5393    5394
                        5490    5491    5583    5584    5697    5698    5777    5778    5813    5814    5835    5905    5930
                        6025    6199    6237    6477    6511    6889    6919    6933    6948    6974    6986    6998    7026
                        7029    7030    7031    7032    7035    7036    7037    7038    7145
DRA   = 000001          1190#
DRDY  = 000200          1197#
DREAD   030774          3157    3264    3362    3492    3625    3756    3887    4018    4150    4270    4375    4473    4604
                        4726    4821    6614#
DRGPSN  030276          3149    3256    3354    3484    3617    3748    3879    4010    4142    4262    4367    4465    4596
                        4718    4813    4954    5058    5183    5274    5375    5472    5565    5678    6458#
DROT  = 000040          1195#
DRPAR = 000010          1174#
DRSYNC  025504          1982    2048    2100    2166    2217    2283    2334    2400    2451    2517    2568    2634    2688
                        2754    2803    2886    2970    3053    3140    3247    3345    3475    3608    3739    3870    4001
                        4133    4253    4358    4456    4587    4709    4804    4841    4945    5049    5174    5265    5366
                        5463    5556    5669    5898#
DRVMSK= 000007          1154#
DSC   = 040000          1201#
DSECTR  025062          1980    2046    2098    2164    2215    2281    2332    2398    2449    2515    2566    2632    2686
                        2752    2801    2884    2968    3051    3138    3245    3343    3473    3606    3737    3868    3999
                        4131    4251    4356    4454    4585    4707    4802    4839    4943    5047    5172    5263    5364
                        5461    5554    5667    5777#
DSWR  = 177570           963#   1308    1837
DTE   = 010000          1183#
DTYE  = 000040          1176#
DT000   041316          1414    8317#
DT0004  041322          1429    1531    1621    1645    1681    1687    8318#
DT0005  041340          1435    8321#
DT0006  041350          1441    1519    8323#
DT0007  041360          1447    1525    8325#
```

```
DT0010  041370          1453    1465    1483    1561    1573    1663    8327#
DT0011  041406          1459    1471    1489    1501    1513    1567    1585    1675    8330#
DT0014  041432          1477    1639    8334#
DT0017  041446          1495    8336#
DT0021  041470          1507    1579    1609    1669    8339#
DT0026  041502          1537    1543    1549    1555    8341#
DT0037  041524          1591    1693    8344#
DT0040  041530          1597    8345#
DT0041  041536          1603    8346#
DT0043  041554          1615    8349#
DT0045  041566          1627    1651    8351#
DT0046  041576          1633    1657    1699    8353#
DT0062  041606          1705    8355#
DT0063  041616          1711    8357#
DWGPSN  027002          1992    2110    2227    2344    2461    2578    2698    2813    2896    2980    3063    6180#
DWRITE  027402          1999    2117    2234    2351    2468    2585    2705    2820    2903    2987    3070    6285#
DWRTCK  030774          4962    5066    5191    5282    5383    5480    5573    5686    6613#
ECCGEN  032260          3378    3508    3641    3772    3903    4034    4166    4489    4620    5082    6316    6663    6694
                        6716    6841#
ECCHI   002340          1786#   6292*   6352    6633*   6698    6849    6855    6860*   6868    6869*   6870*   6871
ECCLO   002342          1787#   6291*   6380    6634*   6699    6859*   6864    6865*   6866*
ECCSRC  002344          1788#   3115*   3221*   3222*   3328*   3458*   3591*   3724*   3854*   3985*   4116*   4228*   4332*
                        4333*   4439*   4571*   4684*   4779*   4913*   4914*   5031*   5143*   5234*   5339*   5436*   5528*
                        5651*   6696
ECCW    = 020000        1217#   5835    5905    5930    6025    6199    6237    6335    6373    6477    6511    6725    6728
ECCXOR  002336          1785#   6851*   6857*   6861
ECH     = 000100        1177#   3415    3545    3678    3809    3940    3942    4071    4073    4203    4526    4657    4659
                        5119
ECPATX  002346          1789#   3155*   3262*   3360*   3490*   3623*   3754*   3885*   4016*   4148*   4268*   4373*   4471*
                        4602*   4724*   4819*   4960*   5064*   5189*   5280*   5381*   5478*   5571*   5684*   6741
ECPOSX  002350          1790#   3156*   3263*   3361*   3381*   3382    3394*   3402*   3491*   3511*   3512    3524*   3532*
                        3624*   3644*   3645    3657*   3665*   3755*   3775*   3776    3788*   3796*   3886*   3906*   3907
                        3919*   3927*   4017*   4037*   4038    4050*   4058*   4149*   4169*   4170    4182*   4190*   4269*
                        4374*   4472*   4492*   4493    4505*   4513*   4603*   4623*   4624    4636*   4644*   4725*   4820*
                        4961*   5065*   5085*   5086    5098*   5106*   5190*   5281*   5382*   5479*   5572*   5685*   6733
                        6746*   6749
EH026   001602          1536#   2924*   3091*
EH030   001622          1548#   2838*   3005*
EH031   001632          1554#   2842*   3009*
EMTVEC= 000030          1052#   1821*   1822*
EMW     001510          1493#   1991*   1998*   2109*   2116*   2226*   2233*   2343*   2350*   2460*   2467*   2577*   2584*
                        2697*   2704*   2812*   2819*   2895*   2902*   2979*   2986*   3062*   3069*   6894*   6924*   6938*
                        6953*   6979*   6991*   7003*
EMW1    045452          6894    8860#
EMW2    045540          6924    6979    8869#
EMW3    045630          6938    6991    8879#
EMW4    045716          6953    7003    8888#
EM000   044512          1412    8771#
EM5004  044557          1427    1433    1439    1445    1631    8778#
EM5010  044771          1451    1457    1487    8806#
EM5011  045051          1463    1469    1475    8815#
EM5015  045154          1481    8827#
EM5020  045211          1499    1991    2109    2226    2343    2460    2577    2697    2812    2895    2979    3062    8832#
EM5021  045270          1505    1511    1998    2116    2233    2350    2467    2584    2704    2819    2902    2986    3069
                        8840#
EM5023  045366          1517    1523    1529    8851#
```

```
EM5026  046006          1535    8898#
EM5027  046146          1541    8915#
EM5030  046320          1547    8933#
EM5031  046413          1553    8943#
EM5032  046500          1559    1565    8952#
EM5034  046556          1571    1583    1679    8960#
EM5035  046623          1577    8967#
EM5037  046664          1589    8973#
EM5041  046740          1601    8981#
EM5042  047025          1607    8990#
EM5043  047106          1613    8999#
EM5044  047165          1619    1625    9007#
EM5046  047235          9014#
EM5047  047327          1637    9024#
EM5050  047422          1643    9034#
EM5051  047475          1649    9042#
EM5052  047552          1655    9050#
EM5053  047672          1661    1673    1685    9064#
EM5054  047761          1667    9074#
EM5060  050024          1691    9080#
EM5061  050103          1697    1703    9088#
EM5063  050160          1709    9096#
ERRCNT  002314          1776#   1854*   7628*   7631
ERRVEC= 000004          1045#   1835    1836*   1847*   7229*   7230*   7250*   7251*   7277    7278    7282*   7287*   7299*
                        7304*   7313*   7323*   7324*   7355    7356*   7358*   7361*
E.ASOF  002236          1741#
E.BA    002224          1736#   4740*   4887*   4888    5296*   5711*   5712*   5713    8353
E.CS1   002220          1734#   2030*   2031    2148*   2149    2265*   2266    2382*   2383    2499*   2500    2616*   2617
                        2736*   2737    3283*   3284*   3285    3408*   3409*   3410*   3411    3538*   3539*   3540*   3541
                        3671*   3672*   3673*   3674    3802*   3803*   3804*   3805    3933*   3934*   3935*   3936    4064*
                        4065*   4066*   4067    4196*   4197*   4198*   4199    4393*   4394*   4395*   4396    4519*   4520*
                        4521*   4522    4650*   4651*   4652*   4653    4750*   4751*   4752*   4753    4866*   4867    4880*
                        4881*   4882    4980*   4981*   4982*   4983    5112*   5113*   5114*   5115    5209*   5210*   5211*
                        5212    5306*   5307*   5308*   5309    5834*   5838    5955*   5956    6114*   6115    6226*   6227
                        6257    6398*   6400    6504*   6505    6549    6777*   6778    8318    8330
E.CS2   002230          1738#
E.DA    002226          1737#   4855*   6120*   6121*   6126    6128*   6129*   6134    6137*   6142    6147    8334
E.DB    002242          1743#
E.DCYL  002240          1742#   4854*   6119*   6138*   6140    6146    8334
E.DS    002232          1739#
E.ECPS  002252          1747#   3382*   3386    3512*   3516    3645*   3649    3776*   3780    3907*   3911    4038*   4042
                        4170*   4174    4493*   4497    4624*   4628    5086*   5090    6733*   6736    6749*   6753    8346
                        8349
E.ECPT  002254          1748#   3383    3513    3646    3777    3908    4039    4171    4494    4625    5087    6318    6665
                        6718    6734    6750    6871*   6872*   8339    8346
E.ER    002234          1740#   3417*   3547*   3680*   3811*   3942*   4073*   4205*   4528*   4659*   4861*   5121*   8351
E.MR1   002244          1744#   5835*   5849    5905*   5909    5930*   5934    5946    6025*   6035    6108*   6199*   6204
                        6237*   6335*   6373*   6477*   6482    6511*   6516    6525*   6528    6539*   6540    6657    6711
                        6725*   6728*   6729*   6888*   6891    6902*   6903*   6908*   6913*   6918*   6921    6932*   6935
                        6946*   6947*   6950    6967*   6970*   6973*   6976    6985*   6988    6997*   7000    8321    8327
                        8336
E.MR2   002246          1745#   2012*   2036    2130*   2154    2247*   2271    2364*   2388    2481*   2505    2598*   2622
                        2718*   2742    5820*   5857    8323
E.MR3   002250          1746#   2026*   2041    2144*   2159    2261*   2276    2378*   2393    2495*   2510    2612*   2627
                        2732*   2747    5832*   5865    8325
E.SPAR  002256          1749#
```

```
E.WC    002222          1735#   5718*   5719    8355
E5004A  044623          1428    1458    1470    1488    1500    1512    1530    1566    1584    1620    1644    1674    1680
                        1686    8785#
E5004B  044642          1434    1452    1464    1482    1560    1572    1662    8788#
E5004C  044661          1440    1518    8791#
E5004D  044700          1446    1524    8794#
E5004E  044717          1626    1650    8797#
E5004F  044735          1698    8800#
E5004G  044753          1704    1710    8803#
E5011A  045124          1476    1638    8823#
E5021A  045336          1506    1578    1668    8847#
E5026A  046061          2838    2842    2924    8906#
E5026B  046232          3005    3009    3091    8924#
FMTE  = 000020          1175#
FSBLVV  034242          2011    2025    2129    2143    2246    2260    2363    2377    2480    2494    2597    2611    2717
                        2731    5819    5831    7205#
GETREG  034154          2832    2915    2999    3082    3282    3407    3537    3670    3801    3932    4063    4195    4392
                        4518    4649    4749    4865    4879    4979    5111    5208    5305    5398    5495    5588    5702
                        5836    5952    6112    6224    6255    6395    6502    6547    6776    7169#
GNS   = ****** U        1237    1874    5748    5755    8300    8301    8302    8303    8304    8306    8308    8309    8310
                        8311    8312    8313    8314
GO    = 000001          1138#   3283    3410    3540    3673    3804    3935    4066    4198    4394    4521    4652    4751
                        4880    4981    5114    5210    5307
GTSWR = 104406          1869    8306#
HIBITS  002354          1792#   3116*   3223*   3310*   3440*   3573*   3706*   3836*   3967*   4098*   4229*   4334*   4421*
                        4553*   4685*   4780*   4910*   5007*   5142*   5233*   5338*   5435*   5537*   5595    5604    5606*
                        5626*   6680
HT    = 000011          955#    7697    7738
HVRC  = 000400          1179#
IBUFF   050440          3121    3168    3228    3315    3445    3578    3711    3841    3972    4103    4234    4281    4339
                        4426    4558    4690    4737    4740    4785    4887    4919    4929    5012    5022    5148    5158
                        5239    5249    5293    5296    5344    5350*   5351    5402    5408*   5441    5447*   5448    5499
                        5505*   5506*   5533    5538*   5592    5632    5642    5706    5711    7057    9192#
IDAE  = 002000          1181#
IE    = 000100          1139#
ILF   = 000001          1171#
INTR  = 000300          1134#
IOTVEC= 000020          1050#   1819*   1820*
IR    = 000100          1158#
KIPAR0= 172340          1083#   7263    7291
KIPAR1= 172342          1084#
KIPAR2= 172344          1085#
KIPAR3= 172346          1086#
KIPAR4= 172350          1087#
KIPAR5= 172352          1088#
KIPAR6= 172354          1089#
KIPAR7= 172356          1090#   7307
KIPDR0= 172300          1072#
KIPDR1= 172302          1073#
KIPDR2= 172304          1074#
KIPDR3= 172306          1075#
KIPDR4= 172310          1076#
KIPDR5= 172312          1077#
KIPDR6= 172314          1078#
KIPDR7= 172316          1079#
LF    = 000012          956#    7732    7738
```

```
LOADRK  034026          1961    2079    2196    2313    2430    2547    2667    2782    2865    2949    3032    3117    3224
                        3311    3441    3574    3707    3837    3968    4099    4230    4335    4422    4554    4686    4781
                        4915    5008    5144    5235    5340    5437    5529    5628    7141#
L.ASOF  002272          1759#   7146*   7161
L.BA    002264          1754#   7143*   7156
L.CS1   002260          1752#   2021    2030    2139    2148    2256    2265    2373    2382    2490    2499    2607    2616
                        2727    2736    3408    3538    3671    3802    3933    4064    4196    4393    4519    4650    4750
                        4866    4980    5112    5209    5306    5802    5808    5828    5834    5955    6008    6090    6093
                        6114    6123    6131    6226    6328    6339    6398    6504    6636    6675    6777    6843    7148*
                        7162
L.CS2   002270          1758#   7147*   7159
L.DA    002266          1755#   4855    6120    6147*   6185    6463    7142*   7158
L.DCYL  002274          1760#   2010    2128    2245    2362    2479    2596    2716    4854    5818    6007    6119    6146*
                        7141*   7157
L.DS    002266          1756#   2019    2137    2254    2371    2488    2605    2725    5827    6005    6098    6102    6190
                        6468
L.DT    002267          1757#   2013    2131    2248    2365    2482    2599    2719    5821    5998
L.MR1   002276          1761#   7145*   7160
L.MR2   002300          1762#
L.MR3   002302          1763#
L.WC    002262          1753#   7144*   7155
MCLK  = 000400          1212#   2005    2123    2240    2357    2474    2591    2711    2827    2910    2994    3077    3276
                        4387    4744    4832    4974    5203    5300    5393    5490    5583    5697    5813    6888    6889
                        6902    6932    6933    6947    6970    6985    6986    6997    7026    7029    7031    7035    7037
MDS   = 001000          1161#
MEMBAS= 172100          1093#   7233
MEMERR  033524          1256    7045#
MEMPAR  002332          1783#   7232*   7238*   7252
MEMVEC= 000114          1092#   1255    7254*   7255*
MERD  = 001000          1213#   7029    7037
MESMSK= 000017          1206#
MEWD  = 002000          1214#   5835    5905    5930    6025    6199    6237    6477    6511    6888    6903    6908    6932
                        6946    6997
MIND  = 000200          1211#
MMVEC = 000250          1061#
MSP   = 000100          1210#   5777
MI.BIT  002322          1779#   4874*   5904*   5942*   6028*   6201*   6233*   6307    6308*   6355    6356*   6385    6386*
                        6479*   6649*   6693*   6704*   6845    6906    6965    6971    7024    8327    8330    8336
M2.BIT  002324          1780#   6234*   6307*   6355*   6385*   6911    6916    8336
NED   = 010000          1164#
NEM   = 004000          1163#
NEWPAS  003376          1931#   5771
NXF   = 000004          1173#
OBUFF   051444          1965    2083    2200    2317    2434    2551    2671    2786    2869    2953    3036    3167    3326*
                        3327*   3329    3331*   3333*   3456*   3457*   3459    3461*   3463*   3589*   3590*   3592    3594*
                        3596*   3722*   3723*   3725*   3726*   3728*   3852*   3853*   3855*   3856*   3858*   3859*   3983*
                        3984*   3986*   3987*   3989*   3990*   4114*   4115*   4117    4119*   4121*   4280    4437*   4438*
                        4440    4442*   4444*   4569*   4570*   4572*   4573*   4575*   4576*   4930    5023    5029*   5030*
                        5032    5034*   5036*   5159    5250    5349*   5353*   5403    5446*   5450*   5500    5539*   5593
                        5643    5649*   5650*   5652    5654*   5656*   5707    6294    6631    6698*   6699*   7073    7113
                        9193#
OFFSET= 000015          1127#
OFST  = 000004          1192#
OPI   = 020000          1184#
OPROO1  042500          1881    8574#
OPROO2  042532          1884    1894    1911    8579#
```

```
OPR003  042540          1891    8580#
OPR004  042570          1902    8584#
OPR006  042612          1878    8587#
OPR007  042661          7650    8594#
OPSTRT  034072          1972    2090    2207    2324    2441    2558    2678    2793    2876    2960    3043    3131    3238
                        3336    3466    3599    3730    3861    3992    4124    4244    4349    4447    4578    4700    4795
                        4936    5040    5165    5256    5357    5454    5547    5660    7155#
OR      = 000200        1159#
PACK    = 000003        1122#
PARCHK  034314          1933    7229#   8261
PARM    002360          1244    1796#
PAR.EN= 000001          1095#   7236
PAT     = 000020        1208#
PAT2    050240          7107    9110#
PAT3    050300          7105    9130#
PAT4    050340          7101    9150#
PAT6    050400          7090    9172#
PCA     = 004000        1215#   6902    6918    6967
PCD     = 010000        1216#   6902    6913    6973
PGE     = 002000        1162#
PIP     = 020000        1200#
PIRQ    = 177772         962#
PIRQVE= 000240          1056#
PR.BIT  002320          1778#   4873*   5903*   5942    5943*   6028    6029*   6032*   6200*   6236*   6308    6309*   6356
                        6357*   6386    6387*   6478*   6537*   6649    6650*   6653*   6693    6704    6705*   6708*   6904
                        7022    8327    8330    8336
PR0     = 000000         979#   1934
PR1     = 000040         980#
PR2     = 000100         981#
PR3     = 000140         982#
PR4     = 000200         983#
PR5     = 000240         984#   1770
PR6     = 000300         985#
PR7     = 000340         986#   1257    1806    7230    8244    8258
PS      = 177776         959#    960
PSW     = 177776         960#
PWRVEC= 000024          1051#   1825*   1826*   8243*   8244*   8257*   8258*
P1.BIT  002316          1777#   6235*   6250*   6309    6310*   6313*   6357    6358*   6361*   6387    6388*   6842    6909
                        6963    8336
RDBIT   033414          3377    3406    3507    3536    3640    3669    3771    3800    3902    3931    4033    4062    4165
                        4194    4488    4517    4619    4648    4875    5081    5110    5907    5932    5944    6033    6202
                        6480    6514    6526    6538    6655    6709    7022#
RDCHR = 104410          8083    8309#
RDDATA= 000021          1129#   3123    3230    3317    3447    3580    3713    3843    3974    4105    4236    4341    4428
                        4560    4692    4787
RDGATE= 100000          1219#   5930    6025    6108    6525    6729
RDHEAD= 000025          1131#
RDLIN = 104411          8157    8310#
RDOCT = 104412          1885    1895    1912    8311#
RDY     = 000200        1140#   3284    3409    3539    3672    3803    3934    4065    4197    4395    4520    4651    4752
                        4881    4982    5113    5211    5308
RECAL = 000013          1126#
RESREG= 104414          7627    8313#
RESTRT  002370          1242    1799#
RESVEC= 000010          1046#
RKASOF= 000016          1109#   7161*
```

```
RKBA   = 000004          1104#  7156*
RKCS1  = 000000          1102#  1957*  2029   2075*  2147   2192*  2264   2309*  2381   2426*  2498.  2543*  2615
                         2660*  2735   2780*  2863*  2945*  3030*  3113*  3220*  3281   3308*  3438*  3571*  3704*
                         3834*  3965*  4096*  4226*  4331*  4419*  4551*  4682*  4776*  4935*  5039*  5164*  5255*
                         5356*  5453*  5541*  5659*  7162*
RKCS2  = 000010          1106#  7159*  7646*
RKDA   = 000006          1105#  7158*
RKDB   = 000024          1111#
RKDCYL= 000020           1110#  7157*
RKDS   = 000012          1107#
RKECPS= 000030           1115#  3380   3510   3643   3774   3905   4036   4168   4491   4622   5084   6732   6748
RKECPT= 000032           1116#  3379   3509   3642   3773   3904   4035   4167   4490   4621   5083   6317   6664
                         6717   6731   6747
RKER   = 000014          1108#  4858
RKMR1  = 000026          1112#  2005*  2006*  2123*  2124*  2240*  2241*  2357*  2358*  2474*  2475*  2591*  2592*
                         2711*  2712*  2827*  2828*  2910*  2911*  2994*  2995*  3077*  3078*  3276*  3277*  4387*
                         4388*  4744*  4745*  4832*  4833*  4974*  4975*  5203*  5204*  5300*  5301*  5393*  5394*
                         5490*  5491*  5583*  5584*  5697*  5698*  5777*  5778*  5813*  5814*  5908   5933   5945
                         6034   6203   6481   6515   6527   6539   6656   6710   6889*  6890   6919*  6920   6933*
                         6934   6948*  6949   6974*  6975   6986*  6987   6998*  6999   7026*  7029*  7030*  7031*
                         7032*  7035*  7036*  7037*  7038*  7160*
RKMR2  = 000034          1113#  2027   2145   2262   2379   2496   2613   2733
RKMR3  = 000036          1114#  2028   2146   2263   2380   2497   2614   2734
RKPRI    002306          1770#  1928*
RKSPAR= 000022           1117#
RKVEC    002304          1769#  1926*  1927*
RKWC   = 000002          1103#  7155*
RLS    = 000010          1155#
SAVREG= 104413           7574   8312#
SAVSWR   002356          1793#  8242*  8262
SCLR   = 000040          1157#
SCOP1  = 104415          5406   5503   5602   8314#
SCOP1$   035374          7446#  8314
SEEK   = 000017          1128#
SELDRV= 000001           1121#
SKI    = 000002          1172#
SPACE2   042724          7610   7622   8600#
SPAR   = 020000          1147#
SPDLSS= 000020           1194#
SRTFLG   002312          1772#  1796*  1799*  1802*  1879
SRTSPL= 000011           1125#
SR0    = 177572          1065#  7288   7303*  7310*
SR1    = 177574          1066#
SR2    = 177576          1067#
SR3    = 172516          1068#  7300*
STACK  = 001100           950#  1805   1817   5881   7013   7053   7638   7649   8259
START    002400          1241   1802#
START1   002404          1797   1800   1803#  7657
STKLMT= 177774            961#
SVAL   = 100000          1202#
SWR      001140          1308#  1815   1837*  1839   1845*  1852*  1867   5842   5851   5859   5867   5872   5915
                         5962   6041   6152   6214   6263   6417   6427   6492   6555   6804   6821   6898   6928
                         6942   6957   6983   6995   7007   7049   7350   7364   7367   7382   7389   7446   7528
                         7535   7547   7551   7629   7927   7966   8021*  8242   8262*
SWREG    000176          1239#  1845   1867   7927   7966   7989
SWO    = 000001          1014#
```

```
SW00   = 000001          1004#   1014
SW01   = 000002          1003#   1013
SW02   = 000004          1002#   1012
SW03   = 000010          1001#   1011
SW04   = 000020          1000#   1010
SW05   = 000040           999#   1009
SW06   = 000100           998#   1008
SW07   = 000200           997#   1007
SW08   = 000400           996#   1006
SW09   = 001000           995#   1005
SW1    = 000002          1013#
SW10   = 002000           994#
SW11   = 004000           993#
SW12   = 010000           992#   7629
SW13   = 020000           991#
SW14   = 040000           990#
SW15   = 100000           989#
SW2    = 000004          1012#
SW3    = 000010          1011#
SW4    = 000020          1010#
SW5    = 000040          1009#
SW6    = 000100          1008#
SW7    = 000200          1007#
SW8    = 000400          1006#
SW9    = 001000          1005#   6898    6928    6942    6957    6983    6995    7007    7049    7446
S.CLR  = 000400          1227#
S.FMT  = 001000          1228#
S.PACK= 004000           1230#
S.RECL= 000040           1224#
S.RTC  = 000200          1226#
S.SEEK= 000020           1223#
S.STSP= 000100           1225#
S.UNLD= 002000           1229#
TBITVE= 000014           1047#
TKVEC  = 000060          1054#   7904#   7905#
TPVEC  = 000064          1055#
TRAPPC  002310           1771#   7046#   8317
TRAPVE= 000034           1053#   1823#   1824#
TRTVEC= 000014           1048#
TSTBY1  043014           8610#
TSTBY2  043024           8612#
TST1    003424           1935    1954#   7409
TST10   007142           2740    2745    2750    2776#   7416
TST11   007412           2859#   7417
TST12   007654           2942#   7418
TST13   010124           3026#   7419
TST14   010366           3109#   7420
TST15   010702           3216#   7421
TST16   01117            3304#   7422
TST17   011722           3435#   7423
TST2    004036           2034    2039    2044    2072#   7410
TST20   012450           3567#   7424
TST21   013176           3699#   7425
TST22   013724           3830#   7426
TST23   014460           3961#   7427
TST24   015214           4092#   7428
```

```
TST25   015742          4222#   7429
TST26   016256          4327#   7430
TST27   016550          4415#   7431
TST3    004450          2152    2157    2162    2189#   7411
TST30   017276          4547#   7432
TST31   020032          4678#   7433
TST32   020346          4773#   7434
TST33   021030          4863    4870    4885    4908#   7435
TST34   021344          5003#   7436
TST35   022114          5138#   7437
TST36   022424          5229#   7438
TST37   022762          5334#   7439
TST4    005062          2269    2274    2279    2306#   7412
TST40   023312          5431#   7440
TST41   023646          5524#   7441
TST42   024222          5623#   7442
TST5    005474          2386    2391    2396    2423#   7413
TST6    006106          2503    2508    2513    2540#   7414
TST7    006520          2620    2625    2630    2657#   7415
TYPDS = 104405          5752    5759    8304#
TYPE  = 104401          1861    1878    1881    1884    1891    1894    1902    1911    5746    5753    5760    7530    7538
                        7586    7587    7591    7592    7599    7610    7612    7622    7623    7625    7633    7650    7702
                        7798    7873    7921    7933    7987    7988    7991    8002    8012    8023    8042    8090    8096
                        8101    8105    8110    8111    8113    8116    8120    8186    8188    8260    8300#
TYPERR  036066          7537    7574#
TYPOC = 104402          1883    1910    7607    7990    8301#
TYPON = 104404          8303#
TYPOS = 104403          8302#
T.ASOF  002176          1722#
T.BA    002164          1717#   4736    4888    5292    5713    8353
T.CS1   002160          1715#   2029*   2031    2147*   2149    2264*   2266    2381*   2383    2498*   2500    2615*   2617
                        2735*   2737    2833    2919    3000    3086    3281*   3285    3411    3541    3674    3805    3936
                        4067    4199    4396    4522    4653    4753    4867    4882    4983    5115    5212    5309    5838
                        5956    6115    6227    6257    6400    6505    6549    6778    7174    8318    8330    8341
T.CS2   002170          1719#   5399    5401    5496    5498    5589    5591    5703    5705    8318    8330    8341
T.DA    002166          1718#   6142    8334    8341
T.DB    002202          1724#
T.DCYL  002200          1723#   6140    8334    8341
T.DS    002172          1720#   8318    8330    8341
T.ECPS  002212          1728#   3380*   3386    3510*   3516    3643*   3649    3774*   3780    3905*   3911    4036*   4042
                        4168*   4174    4491*   4497    4622*   4628    5084*   5090    6732*   6736    6748*   6753    7183*
                        8346    8349
T.ECPT  002214          1729#   3379*   3383    3509*   3513    3642*   3646    3773*   3777    3904*   3908    4035*   4039
                        4167*   4171    4490*   4494    4621*   4625    5083*   5087    6317*   6318    6664*   6665    6717*
                        6718    6731*   6734    6747*   6750    7184*   8339    8346
T.ER    002174          1721#   2836    2916    3002    3083    3415    3545    3678    3809    3940    4071    4203    4526
                        4657    4858*   4859    5119    8318    8330    8341    8351
T.MR1   002204          1725#   5849    5908*   5909    5933*   5934    5945*   5946    6034*   6035    6203*   6204    6481*
                        6482    6515*   6516    6527*   6528    6540    6656*   6657    6710*   6711    6890*   6891    6920*
                        6921    6934*   6935    6949*   6950    6975*   6976    6987*   6988    6999*   7000    8321    8327
                        8336
T.MR2   002206          1726#   2027*   2036    2145*   2154    2262*   2271    2379*   2388    2496*   2505    2613*   2622
                        2733*   2742    5857    7185*   8323
T.MR3   002210          1727#   2028*   2041    2146*   2159    2263*   2276    2380*   2393    2497*   2510    2614*   2627
                        2734*   2747    5865    7186*   8325
T.SPAR  002216          1730#
```

```
T.WC    002162          1716# 5719  8341  8355
UFE   = 000400          1160#
UNLOAD= 000007          1124#
UNS   = 040000          1185#
UPE   = 020000          1165#
VV    = 000100          1196#
WCE   = 040000          1166# 5399  5496  5589  5703
WLE   = 004000          1182#
WRDATA= 000023          1130# 1967  2085  2202  2319  2436  2553  2673  2788  2871  2955  3038
WRDCNT  002330          1782#
WRHEAD= 000027          1132#
WRL   = 004000          1199#
WRTBIT  032450          6240  6251  6314  6362  6389  6888#
WRTCHK= 000031          1133# 4921  5014  5150  5241  5346  5443  5535  5634
WRTGAT= 040000          1218# 6237  6903
WR.PAR= 000004          1094#
$APTHD  001000          1268  1274#
$ASTAT= ****** U        7486  7501
$ATYC   035444          7457  7459#
$ATY1   035420          7455#
$ATY3   035426          7456# 7687
$ATY4   035436          7458# 7543
$AUTOB  001134          1305# 1871* 7980  8137
$BASE   001324          1392# 1882  1890* 1931  7645  8263
$BDADR  001122          1300#
$BDDAT  001126          1302#
$BELL   001240          1343# 7530  7563  7933  8130
$CHARC  036722          7704# 7714* 7721  7730* 7735#
$CKSWR  037676          7966# 8308
$CMTAG  001100          1288# 1812  1813  1821  1827  1828  1829
$CM1  = 000010          1320# 1321# 1322# 1323# 1324# 1325# 1326# 1327# 1328#
$CM2  = 000020          1320# 1321# 1322# 1323# 1324# 1325# 1326# 1327# 1328#
$CM3  = 000010          1318# 1320
$CM4  = 000015          1328# 1329# 1330# 1331# 1332# 1333# 1334# 1335# 1336# 1337# 1338# 1339# 1340#
                        1341#
$CNTLC  040574          7921  8002  8130#
$CNTLG  040606          7987  8132#
$CNTLU  040601          8012  8105  8131#
$CORE   034656          7286  7313#
$CPUOP  001276          1366#
$CRLF   001245          1345# 5760  7538  7563  7586  7591  7599  7612  7625  7703  7738  8023  8110
                        8130  8191
$CROUT  034706          7313  7320#
$DBLK   037370          7839  7873  7881#
$DEVCT  001260          1357#
$DEVM   001326          1393#
$DOAGN  025052          5742  5763  5769#
$DTBL   037360          7842  7877#
$ENDAD  025042          1251  1859  5765# 7558
$ENDCT  024706          1827  5744#
$ENULL  025056          5772#
$ENV    001270          1362# 1865  7464  7488  7540  7682
$ENVM   001271          1363# 1850  7466  7684  7689
$EOP    024652          5734# 7639  7654
$EOPCT  024700          1827* 5741# 5745  7637* 7653*
$ERFLG  001103          1291# 1804* 7339  7378  7380  7386* 7408  7448  7525* 7563  7648*
```

```
$ERMAX  001115          1297#   1830*   7380    7403*   7408
$ERROR  035666          1821    7523#
$ERRPC  001116          1298#   7532*   7533*   7534    7563    8316    8321    8323    8325    8327    8330    8334    8336
                        8339    8341    8344    8346    8349    8351    8353    8355    8357
$ERRTB  001330          1410#   7583
$ERTTL  001112          1295#   5757    5761*   7531*   7563
$ESCAP  001236          1342#   1829*   1932*   3171*   4284*   5840*   5853*   5861*   5869*   5875*   5880*   5901*   5912*
                        5914*   5919*   5920*   5936*   5949*   5964*   6037*   6040*   6045*   6046*   6154*   6206*   6213*
                        6218*   6219*   6265*   6406*   6420*   6426*   6431*   6432*   6484*   6491*   6496*   6497*   6519*
                        6530*   6542*   6557*   6784*   6788*   6792*   6796*   6800*   6808*   6820*   6826*   6828*   6893*
                        6923*   6937*   6952*   6959*   6978*   6990*   7002*   7009*   7045*   7048*   7402*   7554    7556
                        7563    7647*
$ETABL  001270          1361#
$ETEND  001330          1280    1394#
$FATAL  001252          1354#   7492*
$FFLG   035664          7455*   7458*   7486    7495*   7503#
$FILLC  001156          1316#   7707    7738
$FILLS  001155          1315#   7738
$GDADR  001120          1299#
$GDDAT  001124          1301#
$GET42  025032          5762#
$GTSWR  037766          7988#   8306
$HD   = 000000           935
$HIBTS  001000          1275#
$HIOCT  040774          8179#   8190#
$ICNT   001104          1292#   7393*   7394    7396*   7407
$INTAG  001135          1306#   7985*   8004    8024    8137
$ITEMB  001114          1296#   7534*   7542    7563    7577
$KTNEX  034650          7287    7312#
$KTOUT  034640          7304    7309#
$KT11   034506          7285#   7289*   7312*
$LF     001246          1346#   7563    7738    8120    8130    8191
$LFLG   035663          7496*   7502#
$LPADR  001106          1293#   1831*   7375*   7384*   7400*   7405    7407    8264
$LPERR  001110          1294#   1832*   5354*   5451*   5540*   5882    7014    7054    7384    7401*   7407    7450    7553
$LSTAD  034752          7325*   7332#
$LSTBK  034754          7326*   7333#
$MADR1  001302          1379#
$MADR2  001306          1383#
$MADR3  001312          1386#
$MADR4  001316          1389#
$MAIL   001250          1276    1280    1352#   1849    1865    7399    7540    7682
$MAMS1  001300          1373#
$MAMS2  001304          1381#
$MAMS3  001310          1384#
$MAMS4  001314          1387#
$MBADR  001002          1276#
$MFLG   035662          7456*   7462    7497*   7501#
$MNEW   040624          7991    8135#
$MSGAD  001264          1359#   7472*   7475
$MSGLG  001266          1360#   7477*
$MSGTY  001250          1353#   7470    7478*   7490    7494*
$MSWR   040613          7988    8133#
$MTYP1  001301          1374#
$MTYP2  001305          1382#
$MTYP3  001311          1385#
```

```
$MTYP4  001315          1388#
$MXCNT  035266          7397    7407#
$NULL   001154          1314#   7709    7738
$NWTST= 000001          1939#   1941    2057#   2059    2175#   2177    2292#   2294    2409#   2411    2526#   2528    2643#
                        2645    2763#   2765    2847#   2849    2929#   2931    3014#   3016    3097#   3099    3204#   3206
                        3289#   3291    3421#   3423    3552#   3554    3685#   3687    3816#   3818    3947#   3949    4078#
                        4080    4210#   4212    4315#   4317    4400#   4402    4533#   4535    4666#   4668    4759#   4761
                        4896#   4898    4988#   4990    5126#   5128    5216#   5218    5315#   5317    5412#   5414    5510#
                        5512    5610#   5612
$OCNT   037150          7770*   7799*   7812#
$OMODE  037152          7765*   7769*   7774    7777*   7788*   7814#
$OVER   035252          7351    7376    7385    7395    7404#
$PASS   001256          1356#   1849*   5738*   5739*   5750    5772    7391    7408
$PASTM  001006          1278#
$POWER  041220          8260    8267#
$PWRCT  041214          8251*   8252*   8253*   8255*   8266#
$PWRDN  041072          1825    8242#   8257
$PWRUP  041120          8243    8251#
$QUES   001244          1344#   7563    7738    8042    8113    8130    8188    8191
$RDCHR  040240          8055#   8309
$RDDEC= ****** U        8312
$RDLIN  040330          8078#   8310
$RDOCT  040636          8152#   8311
$RDSZ = 000010          8071#
$REGAD  001160          1318#
$REG0   001162          1320#   3187*   4300*   5401*   5498*   5591*   5705*   8345    8357
$REG1   001164          1321#   3188*   4301*   5402*   5499*   5592*   5706*   8345    8357
$REG2   001166          1322#   3189*   4302*   5403*   5500*   5593*   5707*   8345    8357
$REG3   001170          1323#   5594*   8357
$REG4   001172          1324#   5595*   5597*   8357
$REG5   001174          1325#
$REG6   001176          1326#
$REG7   001200          1327#
$RESRE  041034          8224#   8313
$RTNAD  025054          5771#
$R2A  = ****** U        8314
$SAVRE  040776          8208#   8312
$SCOPE  034756          1819    7348#
$SETUP= 000137          1796#   1818    1819    1821    1823    1825    1827    1828    1829    1831    1859    1862    5736
                        7349    7524    7550    7558    7925    7930    7931    7961    8137
$SIZE   034450          1929    7273#
$SIZEX  034712          7311    7321#
$STUP = 177777          1796#
$SVLAD  035216          7359    7398#
$SVPC = 000220          1249#   1254
$SWR  = 167400           923#    935     939     940     941     942     943     944     945     946    1341    1342    1343
                        1828    1829    1831    1832    1955    2073    2190    2307    2424    2541    2658    2777    2860
                        2943    3027    3110    3217    3305    3436    3568    3700    3831    3962    4093    4223    4328
                        4416    4548    4679    4774    4909    5004    5139    5230    5335    5432    5525    5624    5731
                        5737    5764    5770    5772    7340    7341    7342    7343    7344    7350    7362    7364    7365
                        7378    7379    7380    7387    7388    7389    7401    7404    7407    7515    7516    7517    7518
                        7519    7528    7535    7547    7551    7563
$SWREG  001272          1364#   1852
$SWRMK= 000000           946     947    7344    7345    7368
$SWOBT  035270          5919    6045    6218    6431    6496    6826    7374    7408#
$TESTN  001254          1355#   5917    6043    6216    6429    6494    6824    7399*   7575*   7576*   8317    8318    8321
```

```
                        8323    8325    8327    8330    8334    8336    8339    8341    8344    8346    8349    8351    8353
                        8355    8357
$TIMES  001234          1341#   1828*   1955*   2073*   2190*   2307*   2424*   2541*   2658*   2777*   2860*   2943*   3027*
                        3110*   3217*   3305*   3436*   3568*   3700*   3831*   3962*   4093*   4223*   4328*   4416*   4548*
                        4679*   4774*   4909*   5004*   5139*   5230*   5335*   5432*   5525*   5624*   5737*   7387*   7394
                        7397*   7407
$TKB    001146          1311#   7885    7906    7917    7942    7970    7997
$TKCNT  037400          7886#   7901*   7931    7948*   8062    8064*
$TKINT  037410          1810    7901#   7922    7983
$TKQEN= 037407          7890#   7956    8067
$TKQIN  037402          7887#   7902*   7903    7954*   7955*   7956    7958*
$TKQOU  037404          7888#   7903*   8065    8066*   8067    8069*
$TKQSR  037406          7889#   7902    7958    8069
$TKS    001144          1310#   7885    7907*   7938*   7940    7946*   7968    7984*   7994    8006*   8026*
$TKSRV  037460          7904    7917#
$TMP0   001202          1328#   1886*   1888    1890    1896*   1898    1901    1913*   1915    1917    1919*   1920*   1921*
                        1922*   1923*   1925
$TMP1   001204          1329#   6024*   6061    6064*   6067    6070*   6073    6077*   6080    6107*
$TMP10  001222          1336#   6628*   6795
$TMP11  001224          1337#   6630*   6799    6803
$TMP12  001226          1338#   6638*   6642*   6688    6714
$TMP13  001230          1339#   6639*   6643*   6723
$TMP14  001232          1340#   6640*   6644*   6726    6739    6744
$TMP2   001206          1330#
$TMP3   001210          1331#   6624*   6819
$TMP4   001212          1332#   6626*   6817
$TMP5   001214          1333#   3375*   3398    3403*   3505*   3528    3533*   3638*   3661    3666*   3769*   3792    3797*
                        3900*   3923    3928*   4031*   4054    4059*   4163*   4186    4191*   4486*   4509    4514*   4617*
                        4640    4645*   5079*   5102    5107*   6303*   6330    6341    6345*   6348*   6351*   6369    6376
                        6378*   6635*   6677    6681*   6683*   6695*   6762    6766*   6769    6773*
$TMP6   001216          1334#   6295*   6409    6412    6415    6620*   6783    6791
$TMP7   001220          1335#   3172*   3191    3193*   4285*   4304    4306*   6297*   6396    6405    6622*   6787
$TN   = 000043           924#    935    1939    1955#   2034    2039    2044    2057    2073#   2152    2157    2162    2175
                        2190#   2269    2274    2279    2292    2307#   2386    2391    2396    2409    2424#   2503    2508
                        2513    2526    2541#   2620    2625    2630    2643    2658#   2740    2745    2750    2763    2777#
                        2847    2860#   2929    2943#   3014    3027#   3097    3110#   3204    3217#   3289    3305#   3421
                        3436#   3552    3568#   3685    3700#   3816    3831#   3947    3962#   4078    4093#   4210    4223#
                        4315    4328#   4400    4416#   4533    4548#   4666    4679#   4759    4774#   4863    4870    4885
                        4896    4909#   4988    5004#   5126    5139#   5216    5230#   5315    5335#   5412    5432#   5510
                        5525#   5610    5624#   7369    7409
$TPB    001152          1313#   7727*   7738
$TPFLG  001157          1317#   7676    7738
$TPS    001150          1312#   7725    7738
$TRAP   041226          1823    8277#
$TRAP2  041250          8288#   8299
$TRP  = 000016          8292#   8301#   8302#   8303#   8304#   8305#   8306    8307#   8308    8309#   8310#   8311#   8312#
                        8313#   8314#   8315#
$TRPAD  041262          8282    8299#
$TSTM   001004          1277#
$TSTNM  001102          1290#   5736*   7339    7371*   7398*   7399    7404    7408    7527    7563    7575
$TTYIN  040564          8080    8081    8093    8111    8125    8129#
$TYPBN= ****** U        8305
$TYPDS  037154          7827#   8304
$TYPE   036444          7483    7676#   8292    8300
$TYPEC  036656          7706    7713    7720    7725#   7726    8028
$TYPEX  036724          7731    7733    7736#
```

```
$TYPOC  036752           7768#  8301
$TYPON  036766           7767   7770#  8303
$TYPOS  036726           7763#  8302
$UNIT   001262           1358#
$UNITM  001010           1279#
$USWR   001274           1365#
$VECT1  001320           1390#  1892   1900*  1901*  1904   1924*  1925*  1926   1928
$VECT2  001322           1391#
$XTSTR  034770           7353#
$$GET4= 000000           5764#
$$SWO8= 000043           7408#  7409   7410#  7411#  7412#  7413#  7414#  7415#  7416#  7417#  7418#  7419#  7420#
                         7421#  7422#  7423#  7424#  7425#  7426#  7427#  7428#  7429#  7430#  7431#  7432#  7433#
                         7434#  7435#  7436#  7437#  7438#  7439#  7440#  7441#  7442#  7443#
$OFILL  037151           7764*  7768*  7778   7813#
$40CAT= ****** U         7350   7537
.       = 052450         1233#  1237#  1243#  1249   1250#  1252#  1254#  1255#  1258#  1264   1265#  1267#  1269#
                         1287#  1347   1816   1831   1832   1875#  5749#  5772   5773#  7407   7408   7504#  7563
                         7738   7881#  7885   7889#  7890   7891#  8129#  8130   8137#  8191   8246   9105#  9192#
                         9193#
.$ASTA= ****** U         7456   7459
.$X     = 001000         1264#  1269
```

BYPASS 1259#
COMMEN 1057#
ENDCOM 1057#
ERROR 951# 1975 1976 1977 1978 1983 1984 1987 1988 1989 1993 1994 1995 1996 2000
2001 2002 2033 2038 2043 2049 2050 2053 2054 2055 2093 2094 2095 2096 2101
2102 2105 2106 2107 2111 2112 2113 2114 2118 2119 2120 2151 2156 2161 2167
2168 2171 2172 2173 2210 2211 2212 2213 2218 2219 2222 2223 2224 2228 2229
2230 2231 2235 2236 2237 2268 2273 2278 2284 2285 2288 2289 2290 2327 2328
2329 2330 2335 2336 2339 2340 2341 2345 2346 2347 2348 2352 2353 2354 2385
2390 2395 2401 2402 2405 2406 2407 2444 2445 2446 2447 2452 2453 2456 2457
2458 2462 2463 2464 2465 2469 2470 2471 2502 2507 2512 2518 2519 2522 2523
2524 2561 2562 2563 2564 2569 2570 2573 2574 2575 2579 2580 2581 2582 2586
2587 2588 2619 2624 2629 2635 2636 2639 2640 2641 2681 2682 2683 2684 2689
2690 2693 2694 2695 2699 2700 2701 2702 2706 2707 2708 2739 2744 2749 2755
2756 2759 2760 2761 2796 2797 2798 2799 2804 2805 2808 2809 2810 2814 2815
2816 2817 2821 2822 2823 2839 2843 2879 2880 2881 2882 2887 2888 2891 2892
2893 2897 2898 2899 2900 2904 2905 2906 2921 2925 2963 2964 2965 2966 2971
2972 2975 2976 2977 2981 2982 2983 2984 2988 2989 2990 3006 3010 3046 3047
3048 3049 3054 3055 3058 3059 3060 3064 3065 3066 3067 3071 3072 3073 3088
3092 3133 3134 3135 3136 3141 3142 3145 3146 3147 3150 3151 3152 3153 3159
3160 3161 3162 3163 3164 3194 3197 3240 3241 3242 3243 3248 3249 3252 3253
3254 3257 3258 3259 3260 3266 3267 3268 3269 3270 3271 3287 3338 3339 3340
3341 3346 3347 3350 3351 3352 3355 3356 3357 3358 3364 3365 3366 3367 3368
3369 3385 3388 3413 3418 3468 3469 3470 3471 3476 3477 3480 3481 3482 3485
3486 3487 3488 3494 3495 3496 3497 3498 3499 3515 3518 3543 3548 3601 3602
3603 3604 3609 3610 3613 3614 3615 3618 3619 3620 3621 3627 3628 3629 3630
3631 3632 3648 3651 3676 3681 3732 3733 3734 3735 3740 3741 3744 3745 3746
3749 3750 3751 3752 3758 3759 3760 3761 3762 3763 3779 3782 3807 3812 3863
3864 3865 3866 3871 3872 3875 3876 3877 3880 3881 3882 3883 3889 3890 3891
3892 3893 3894 3910 3913 3938 3943 3994 3995 3996 3997 4002 4003 4006 4007
4008 4011 4012 4013 4014 4020 4021 4022 4023 4024 4025 4041 4044 4069 4074
4126 4127 4128 4129 4134 4135 4138 4139 4140 4143 4144 4145 4146 4152 4153
4154 4155 4156 4157 4173 4176 4201 4206 4246 4247 4248 4249 4254 4255 4258
4259 4260 4263 4264 4265 4266 4272 4273 4274 4275 4276 4277 4307 4310 4351
4352 4353 4354 4359 4360 4363 4364 4365 4368 4369 4370 4371 4377 4378 4379
4380 4381 4382 4398 4449 4450 4451 4452 4457 4458 4461 4462 4463 4466 4467
4468 4469 4475 4476 4477 4478 4479 4480 4496 4499 4524 4529 4580 4581 4582
4583 4588 4589 4592 4593 4594 4597 4598 4599 4600 4606 4607 4608 4609 4610
4611 4627 4630 4655 4660 4702 4703 4704 4705 4710 4711 4714 4715 4716 4719
4720 4721 4722 4728 4729 4730 4731 4732 4733 4741 4755 4797 4798 4799 4800
4805 4806 4809 4810 4811 4814 4815 4816 4817 4823 4824 4825 4826 4827 4828
4842 4843 4846 4847 4856 4862 4869 4884 4890 4938 4939 4940 4941 4946 4947
4950 4951 4952 4955 4956 4957 4958 4964 4965 4966 4967 4968 4969 4985 5042
5043 5044 5045 5050 5051 5054 5055 5056 5059 5060 5061 5062 5068 5069 5070
5071 5072 5073 5089 5092 5117 5122 5167 5168 5169 5170 5175 5176 5179 5180
5181 5184 5185 5186 5187 5193 5194 5195 5196 5197 5198 5214 5258 5259 5260
5261 5266 5267 5270 5271 5272 5275 5276 5277 5278 5284 5285 5286 5287 5288
5289 5297 5311 5359 5360 5361 5362 5367 5368 5371 5372 5373 5376 5377 5378
5379 5385 5386 5387 5388 5389 5390 5404 5456 5457 5458 5459 5464 5465 5468
5469 5470 5473 5474 5475 5476 5482 5483 5484 5485 5486 5487 5501 5549 5550
5551 5552 5557 5558 5561 5562 5563 5566 5567 5568 5569 5575 5576 5577 5578
5579 5580 5600 5662 5663 5664 5665 5670 5671 5674 5675 5676 5679 5680 5681
5682 5688 5689 5690 5691 5692 5693 5708 5715 5721 7047
ESCAPE 1057#
FSECWT 1764# 1961 2079 2196 2313 2430 2547 2667
GETPRI 1057# 7281

```
GETSWR  1057#  1862#
LDLPER  1259#
MSG     1939#  1941   2057#  2059   2175#  2177   2292#  2294   2409#  2411   2526#  2528   2643#  2645   2763#
        2765   2847#  2849   2929#  2931   3014#  3016   3097#  3099   3204#  3206   3289#  3291   3421#  3423
        3552#  3554   3685#  3687   3816#  3818   3947#  3949   4078#  4080   4210#  4212   4315#  4317   4400#
        4402   4533#  4535   4666#  4668   4759#  4761   4896#  4898   4988#  4990   5126#  5128   5216#  5218
        5315#  5317   5412#  5414   5510#  5512   5610#  5612
MULT    1057#
NEWTST  1057#  1939   2057   2175   2292   2409   2526   2643   2763   2847   2929   3014   3097   3204   3289
        3421   3552   3685   3816   3947   4078   4210   4315   4400   4533   4666   4759   4896   4988   5126
        5216   5315   5412   5510   5610
OSECRD  1765#  3115   3222   3310   3440   3573   3706   3836   3967   4098   4228   4333   4421   4553   4684
        4779
OSECWC  1766#  4914   5008   5143   5234   5339   5436   5628
POP     1057#  5876   5921   6048   6220   6433   6498   6810   7118   7187   7221   7498   7499   7868   8180
        8229
PUSH    1057#  5799   5898   5991   6180   6285   6458   6614   7068   7169   7205   7459   7461   7482   7827
        8154   8209
REPORT  1057#
SCOPE    952#  1954   2072   2189   2306   2423   2540   2657   2776   2859   2942   3026   3109   3216   3304
        3435   3567   3699   3830   3961   4092   4222   4327   4415   4547   4678   4773   4908   5003   5138
        5229   5334   5431   5524   5623   5735
SETPRI  1057#  8058
SETTRA  8292#  8301   8302   8303   8304   8306   8308   8309   8310   8311   8312   8313   8314
SETUP   1057#  1811
SKIP    1057#  2034   2039   2044   2152   2157   2162   2269   2274   2279   2386   2391   2396   2503   2508
        2513   2620   2625   2630   2740   2745   2750   4863   4870   4885
SLASH   1057#
SPACE   1057#
SSECWT  1765#  2782   2865   2949   3032
STARS   1057#  1247   1261   1263   1270   1283   1347   1350   1939   1953   2057   2071   2175   2188   2292
        2305   2409   2422   2526   2539   2643   2656   2763   2775   2847   2858   2929   2941   3014   3025
        3097   3108   3204   3215   3289   3303   3421   3434   3552   3566   3685   3698   3816   3829   3947
        3960   4078   4091   4210   4221   4315   4326   4400   4414   4533   4546   4666   4677   4759   4772
        4896   4907   4988   5002   5126   5137   5216   5228   5315   5333   5412   5430   5510   5523   5610
        5622   5727   7259   7336   7443   7454   7511   7564   7573   7661   7740   7817   7884   7961   7976
        8047   8071   8140   8193   8239   8248   8271
SWRSU   1057#  1833#
TRMTRP  8292#
TYPBIN  1057#
TYPDEC  1057#  5750   5757
TYPNAM  1057#  1855
TYPNUM  1057#
TYPOCS  1057#
TYPOCT  1057#  1882   7989
TYPTXT  1057#  5746   5753
$$CMRE  1281#  1320   1321   1322   1323   1324   1325   1326   1327
$$CMTM  1281#  1328   1329   1330   1331   1332   1333   1334   1335   1336   1337   1338   1339   1340
$$ESCA  1057#
$$NEWT  1057#  1939   2057   2175   2292   2409   2526   2643   2763   2847   2929   3014   3097   3204   3289
        3421   3552   3685   3816   3947   4078   4210   4315   4400   4533   4666   4759   4896   4988   5126
        5216   5315   5412   5510   5610
$$SET   8292#  8301   8302   8303   8304   8306   8308   8309   8310   8311   8312   8313   8314
$$SETM  1849#
$$SKIP  1057#  2034   2039   2044   2152   2157   2162   2269   2274   2279   2386   2391   2396   2503   2508
        2513   2620   2625   2630   2740   2745   2750   4863   4870   4885
```

```
.EQUAT    923#    947
.HEADE    923#    925
.KT11     923#   1057
.SETUP    923#   1796
.SWRHI    923#    935
.SWRLO    923#    947#
.$ACT1    923#   1245
.$APTB   1348#
.$APTH    923#   1259
.$APTY    923#   7452
.$CATC    923#   1231
.$CMTA    923#   1281
.$EOP     923#   5725
.$ERRO    923#   7509
.$ERRT    923#
.$POWE    923#
.$RDOC    923#   8138
.$READ    923#   7882
.$SAVE    923#   8191
.$SCOP    923#   7334
.$SIZE    923#   7257
.$TRAP    923#   8269
.$TYPD    923#   7815
.$TYPE    923#   7659
.$TYPO    923#   7738

. ABS.  052450     000

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

DSKZ:DZR6EA,DSKZ:DZR6EA.SEQ/SOL/DOC/CRF=DZR6EA.P11
RUN-TIME: 86 81 8 SECONDS
RUN-TIME RATIO: 246/177=1.3
CORE USED:  31K  (61 PAGES)

DOCUMENT PAGES:  195
```

Spooler runtime 33 Seconds, 138 KCS, 861 disk reads, 0 disk writes, 195 pages

[redacted] Date 01-Dec-76 17:47:08 Monitor IPC-0 602 [6DD] [redacted]