RK611

DISKLESS CONTROL PART 5

CZR6EB0

AH-9114B-MC

FICHE 1 OF 1

MAR 1978

digital

MADE IN USA

```
.REM    %
                    IDENTIFICATION

PRODUCT CODE:       AC-S112B-MC

PRODUCT NAME:       CZR6EBO RK611 DSKLS CTRL PRT 5

DATE:               FEB 1978

MAINTAINER:         DIAGNOSTIC GROUP

AUTHOR:             MARV TEGROTENHUIS


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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A
LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE
TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR
THE USE OF RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS
NOT SUPLIED BY DIGITAL.

      COPYRIGHT (C) 1976, 1978 BY DIGITAL EQUIPMENT CORPORATION
```

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 (CZR6A)
        RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 2 (CZR6B)
        RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 3 (CZR6C)
        RK611 DISKLESS CONTROLLER DIAGNOSTIC: PART 4 (CZR6D)

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 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 6  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 7  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.

        **NPR WRITING OF MEMORY

                TEST 10 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 11 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 12 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 13 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 14 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 15 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 16 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 17 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 20 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 21 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 22 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.

                TEST 23 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 24 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 25 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 26 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 27 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 30 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 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.


        TEST 31 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 32 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 33 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

                  CONFIGURATION AS WRITE CHECKED DATA:

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

        TEST 34 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 35 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 ERROR.

                                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 O
                        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
                ****************************************************************

                5835    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
                              ;*****************************************************************
```

```
                        ; *** REV 004 ***
                        .NLIST  CND,MD,MC
                        .LIST   ME
                        .ENABL  ABS,AMA
        167400          $SWR=   167400
        000001          $TN=    1
                        .TITLE  CZR6EBO RK611 DSKLS CTRL PRT5
                        ;*COPYRIGHT (C) 1976,1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY MARV TEGROTENHUIS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          ---------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            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
```

```
        000006          SP=     %6                      ;;STACK POINTER
        000007          PC=     %7                      ;;PROGRAM COUNTER

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
        100000          SW15=   100000
        040000          SW14=   40000
        020000          SW13=   20000
        010000          SW12=   10000
        004000          SW11=   4000
        002000          SW10=   2000
        001000          SW09=   1000
        000400          SW08=   400
        000200          SW07=   200
        000100          SW06=   100
        000040          SW05=   40
        000020          SW04=   20
        000010          SW03=   10
        000004          SW02=   4
        000002          SW01=   2
        000001          SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
        100000          BIT15=  100000
        040000          BIT14=  40000
        020000          BIT13=  20000
        010000          BIT12=  10000
        004000          BIT11=  4000
        002000          BIT10=  2000
        001000          BIT09=  1000
        000400          BIT08=  400
        000200          BIT07=  200
        000100          BIT06=  100
        000040          BIT05=  40
        000020          BIT04=  20
        000010          BIT03=  10
        000004          BIT02=  4
```

```
        000002          BIT01=  2
        000001          BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

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

                        ;*KT11 VECTOR ADDRESS

        000250          MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

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

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

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

```
        172350          KIPAR4= 172350
        172352          KIPAR5= 172352
        172354          KIPAR6= 172354
        172356          KIPAR7= 172356

        000114          MEMVEC= 114                     ;MEMORY PARITY VECTOR
        172100          MEMBAS= 172100                  ;MEM PARITY OPTION
        000004          WR.PAR= 4                       ;WRITE BAD PARITY
        000001          PAR.EN= 1                       ;ENABLE PARITY ENABLE
        120210          AVECT1= 120210                  ;DEFINE RK611 VECTOR ADDRESS
        000005          APRIOR= 5                       ;DEFINE RK611 PRIORITY
        177440          ABASE=  177440                  ;DEFINE BASE OF RK611 REGISTERS

                        .SBTTL  RK611 CONTROLLER REGISTER DEFINITION

        000000          RKCS1=  0                       ;CONTROL AND STATUS REGISTER 1
        000002          RKWC=   2                       ;WORD COUNT REGISTER
        000004          RKBA=   4                       ;BUS ADDRESS REGISTER
        000006          RKDA=   6                       ;DESIRED TRACK SECTOR REGISTER
        000010          RKCS2=  10                      ;CONTROL AND STATUS REGISTER 2
        000012          RKDS=   12                      ;DRIVE STATUS REGISTER
        000014          RKER=   14                      ;ERROR REGISTER
        000016          RKASOF= 16                      ;ATTENTION SUMMARY AND OFFSET REGISTER
        000020          RKDCYL=20                       ;DESIRED CYLINDER REGISTER
        000024          RKDB=   24                      ;DATA BUFFER
        000026          RKMR1=  26                      ;MAINTENANCE REGISTER 1
        000034          RKMR2=  34                      ;MAINTENANCE REGISTER 2
        000036          RKMR3=  36                      ;MAINTENANCE REGISTER 3
        000030          RKECPS= 30                      ;ECC POSITION INFORMATION
        000032          RKECPT= 32                      ;ECC PATTERN INFORMATION
        000022          RKSPAR= 22                      ;SPARE REGISTER

                        .SBTTL  DRIVE COMMANDS

        000001          SELDRV= 01                      ;SELECT DRIVE
        000003          PACK=   03                      ;PACK ACKNOWLEDGE
        000005          CLEAR=  05                      ;DRIVE CLEAR
        000007          UNLOAD= 07                      ;UNLOAD
        000011          SRTSPL= 11                      ;START SPINDLE
        000013          RECAL=  13                      ;RECALIBRATE
        000015          OFFSET= 15                      ;OFFSET
        000017          SEEK=   17                      ;SEEK
        000021          RDDATA= 21                      ;READ DATA
        000023          WRDATA= 23                      ;WRITE DATA
        000025          RDHEAD= 25                      ;READ HEADER
        000027          WRHEAD= 27                      ;WRITE HEADER AND DATA
        000031          WRTCHK= 31                      ;WRITE CHECK
        000300          INTR=   300                     ;GENERATE INTERRUPT TO CPU

                        .SBTTL  CONTROL AND STATUS REGISTER 1 BITS

        000001          GO=     BIT0                    ;GO BIT
        000100          IE=     BIT6                    ;INTERRUPT ENABLE
        000200          RDY=    BIT7                    ;CONTROLLER READY
        000400          BA16=   BIT8                    ;BUS ADDRESS BIT 16
        001000          BA17=   BIT9                    ;BUS ADDRESS BIT 17
```

```
        002000                  CDT=    BIT10                   ;CONTROLLER DRIVE TYPE (0=RK06)
        004000                  CTO=    BIT11                   ;CONTROLLER TIMED OUT WAITING FOR
                                                                ;   DRIVE RESPONSE
        010000                  CFMT=   BIT12                   ;CONTROLLER DRIVE FORMAT (0=26 SECTOR, 1=24 SECTOR)
        020000                  SPAR=   BIT13                   ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
        040000                  DI=     BIT14                   ;DRIVE INTERRUPT
        100000                  CERR=   BIT15                   ;CONTROLLER ERROR
        100000                  CCLR=   BIT15                   ;CONTROLLER CLEAR

                                .SBTTL  CONTROL AND STATUS REGISTER 2 BITS

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

                                .SBTTL  ERROR REGISTER BIT DEFINITION

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

                                .SBTTL  STATUS REGISTER BIT DEFINITION

        000001                  DRA=    BIT0                    ;DRIVE AVAILABLE (CONTROLLER IS SET IF
                                                                ;   THIS BIT IS RESET)
        000004                  OFST=   BIT2                    ;DRIVE OFFSET
        000010                  ACLO=   BIT3                    ;AC LOW
        000020                  SPDLSS= BIT4                    ;SPEED LOSS
        000040                  DROT=   BIT5                    ;DRIVE OFF TRACK
        000100                  VV=     BIT6                    ;VOLUME VALID
        000200                  DRDY=   BIT7                    ;DRIVE READY
        000400                  DDT=    BIT8                    ;DRIVE TYPE (0=RK06)
```

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

                                     .SBTTL   MAINTENANCE REGISTER 1 BIT DEFINITION

          000017                     MESMSK= 17                     ;MESSAGE MASK

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

                                     .SBTTL   TRANSMITTED MESSAGE A

          000020                     S.SEEK= BIT4                   ;SEEK COMMAND
          000040                     S.RECL= BIT5                   ;RECALIBRATE COMMAND
          000100                     S.STSP= BIT6                   ;START SPINDLE COMMAND
          000200                     S.RTC=   BIT7                  ;DRIVE RETURN TO CENTERLINE COMMAND
          000400                     S.CLR=   BIT8                  ;CLEAR ERROR AND DSC
          001000                     S.FMT=   BIT9                  ;FORMAT
          002000                     S.UNLD= BIT10                  ;UNLOAD
          004000                     S.PACK= BIT11                  ;SET VOLUME VALID (PACK ACKNOWNLEDGE)
                                     .SBTTL   TRAP CATCHER

          000000                              .=0
                                     ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                     ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                     ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
          000174                              .=174
000174    000000                     DISPREG: .WORD  0              ;;SOFTWARE DISPLAY REGISTER
000176    000000                     SWREG:   .WORD  0              ;;SOFTWARE SWITCH REGISTER
                                     .SBTTL   STARTING ADDRESS(ES)
000200    000137   002400                     JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204    000137   002370                     JMP     RESTRT        ;JUMP TO RESTART ROUTINE
          000214                              .=214
000214    000137   002360                     JMP     PARM          ;JUMP TO OPERATOR ASSIGNED PARMETERS
                                     .SBTTL   ACT11 HOOKS

                                     ;;*****************************************************************
                                     ;HOOKS REQUIRED BY ACT11
          000220                              $SVPC=.               ;SAVE PC
          000046                              .=46
000046    023276                              $ENDAD                ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
          000052                              .=52
000052    000000                              .WORD   0             ;;2)SET LOC.52 TO ZERO
          000220                              .=$SVPC               ;; RESTORE PC
```

```
                000114                  .=MEMVEC
        000114  031726                  MEMERR
        000116  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
        000024  000200                  200       ;;FOR APT START UP
                000044                  .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
        000044  001000                  $APTHDR   ;;POINT TO APT HEADER BLOCK
                001000                  .=.$X     ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

        001000                  $APTHD:
        001000  000000          $HIBTS: .WORD    0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
        001002  001250          $MBADR: .WORD    $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
        001004  000132          $TSTM:  .WORD    90.        ;;RUN TIM OF LONGEST TEST
        001006  001604          $PASTM: .WORD    900.       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
        001010  000144          $UNITM: .WORD    100.       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        001012  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
```

N02


```
001224  000000                  $TMP11: .WORD   0                       ;;USER DEFINED
001226  000000                  $TMP12: .WORD   0                       ;;USER DEFINED
001230  000000                  $TMP13: .WORD   0                       ;;USER DEFINED
001232  000000                  $TMP14: .WORD   0                       ;;USER DEFINED
001234  000000                  $TIMES: 0                               ;;MAX. NUMBER OF ITERATIONS
001236  000000                  $ESCAPE:0                               ;;ESCAPE ON ERROR ADDRESS
001240  177607  000377          $BELL:  .ASCIZ  <207><377><377>         ;;CODE FOR BELL
001244     077                  $QUES:  .ASCII  /?/                     ;;QUESTION MARK
001245     015                  $CRLF:  .ASCII  <15>                    ;;CARRIAGE RETURN
001246  000012                  $LF:    .ASCIZ  <12>                    ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001250                          $MAIL:                          ;;APT MAILBOX
001250  000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001252  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001254  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001256  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
001260  000000                  $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001262  000000                  $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001264  000000                  $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001266  000000                  $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001270                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001270     000                  $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001271     000                  $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001272  000000                  $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001274  000000                  $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001276  000000                  $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                                ;*                              BITS 15-11=CPU TYPE
                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                                      11/70=06,PDQ=07,Q=10
                                ;*                              BIT 10=REAL TIME CLOCK
                                ;*                              BIT  9=FLOATING POINT PROCESSOR
                                ;*                              BIT  8=MEMORY MANAGEMENT
001300     000                  $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
001301     000                  $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
                                ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                                      900 NSEC CORE=001
                                ;*                                      300 NSEC BIPOLAR=002
                                ;*                                      500 NSEC MOS=003
001302  000000                  $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
                                ;*                               MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001304     000                  $MAMS2: .BYTE   AMAMS2          ;;HIGH ADDRESS,M.S. BYTE
001305     000                  $MTYP2: .BYTE   AMTYP2          ;;MEM.TYPE,BLK#2
001306  000000                  $MADR2: .WORD   AMADR2          ;;MEM.LAST ADDRESS,BLK#2
001310     000                  $MAMS3: .BYTE   AMAMS3          ;;HIGH ADDRESS,M.S.BYTE
001311     000                  $MTYP3: .BYTE   AMTYP3          ;;MEM.TYPE,BLK#3
001312  000000                  $MADR3: .WORD   AMADR3          ;;MEM.LAST ADDRESS,BLK#3
001314     000                  $MAMS4: .BYTE   AMAMS4          ;;HIGH ADDRESS,M.S.BYTE
001315     000                  $MTYP4: .BYTE   AMTYP4          ;;MEM.TYPE,BLK#4
001316  000000                  $MADR4: .WORD   AMADR4          ;;MEM.LAST ADDRESS,BLK#4
001320  120210                  $VECT1: .WORD   AVECT1          ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001322  000000                  $VECT2: .WORD   AVECT2          ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001324  177440                  $BASE:  .WORD   ABASE           ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
```

```
1348  001326  000000        $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
1349  001330                $ETEND:
1350                        .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  042704                         EM000                   ;UNEXPECTED PARITY ERROR
 001332  041263                         DH000C
 001334  037510                         DT000
 001336  040026                         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  042751                         EM5004                  ;CS1 IN ERROR
 001362  043015                         ES004A
 001364  037514                         DT0004
 001366  040032                         DF0004

                                ;       ERROR 5                 ;MR1 IN ERROR ATTEMPTING IMPLIED SEEK
 001370  042751                         EM5004
 001372  043034                         ES004B
 001374  037532                         DT0005
 001376  040062                         DF0005

                                ;       ERROR 6                 ;MR2 IN ERROR ATTEMPTING IMPLIED SEEK
 001400  042751                         EM5004
 001402  043053                         ES004C
 001404  037542                         DT0006
 001406  040106                         DF0006

                                ;       ERROR 7                 ;MR3 IN ERROR ATTEMPTING IMPLIED SEEK
 001410  042751                         EM5004
 001412  043072                         ES004D
 001414  037552                         DT0007
 001416  040132                         DF0007

                                ;       ERROR 10                ;MR1 ERROR READING PREAMBLE
 001420  043163                         EM5010
```

```
001422  043034                    ES004B
001424  037562                    DT0010
001426  040156                    DF0010
                  ;               ERROR 11              ;CS1 ERROR READING PREAMBLE
001430  043163                    EM5010
001432  043015                    ES004A
001434  037600                    DT0011
001436  040202                    DF0011
                  ;               ERROR 12              ;MR1 IN ERROR HEADER READ/COMPARE
001440  043243                    EM5011
001442  043034                    ES004B
001444  037562                    DT0010
001446  040156                    DF0010
                  ;               ERROR 13              ;CS1 IN ERROR HEADER READ/COMPARE
001450  043243                    EM5011
001452  043015                    ES004A
001454  037600                    DT0011
001456  040202                    DF0011
                  ;               ERROR 14              ;PACK ADDRESS ERROR HDR READ/COMPARE
001460  043243                    EM5011
001462  043316                    ES011A
001464  037624                    DT0014
001466  040232                    DF0014
                  ;               ERROR 15              ;MR1 ERROR READING GAP
001470  043346                    EM5015
001472  043034                    ES004B
001474  037562                    DT0010
001476  040156                    DF0010
                  ;               ERROR 16              ;CS1 ERROR AFTER READING GAP
001500  043163                    EM5010
001502  043015                    ES004A
001504  037600                    DT0011
001506  040202                    DF0011
                  ;               ERROR 17              ;ERROR IN WRITING
001510  000000    EMW:            0                     ;REGISTER IN ERROR AND CLOCK
001512  000000                    0                     ;TRANSITION IS IDENTIFIED
001514  037640                    DT0017
001516  040256                    DF0017
                  ;               ERROR 20              ;ERROR IN CS1 AFTER WRITING SYNC
001520  043403                    EM5020
001522  043015                    ES004A
001524  037600                    DT0011
001526  040202                    DF0011
                  ;               ERROR 21              ;ERROR IN ECC WORDS
001530  043462                    EM5021
001532  043530                    ES021A
001534  037662                    DT0021
```

```
001536  040302                      DF0021

                    ;               ERROR 22                ;ERR IN CS1 AFTER WRITE DATA
001540  043462                      EM5021
001542  043015                      E5004A
001544  037600                      DT0011
001546  040202                      DF0011

                    ;               ERROR 23                ;ERROR IN MR2
001550  043560                      EM5023                  ;DOING MULTI-SECTOR WRITE
001552  043053                      E5004C
001554  037542                      DT0006
001556  040106                      DF0006

                    ;               ERROR 24                ;MR3 IN ERROR
001560  043560                      EM5023                  ;DOING MULTI-SECTOR WRITE
001562  043072                      E5004D
001564  037552                      DT0007
001566  040132                      DF0007

                    ;               ERROR 25                ;CS1 IN ERROR
001570  043560                      EM5023                  ;AFTER MULTI-SECTOR WRITE
001572  043015                      E5004A
001574  037514                      DT0004
001576  040032                      DF0004

                    ;               ERROR 26
001600  044200                      EM5026                  ;NO COE ERROR
001602  000000      EH026:          .WORD    0
001604  037674                      DT0026
001606  040326                      DF0026

                    ;               ERROR 27
001610  044340                      EM5027                  ;OTHER ERROR WHILE FORCING COE
001612  041232                      DH000A
001614  037674                      DT0026
001616  040326                      DF0026

                    ;               ERROR 30
001620  044512                      EM5030                  ;UNEXPECTED CYL OVERFLOW
001622  000000      EH030:          .WORD    0
001624  037674                      DT0026
001626  040326                      DF0026

                    ;               ERROR 31
001630  044605                      EM5031                  ;UNEXPECTED ERROR TESTING COE
001632  000000      EH031:          .WORD    0
001634  037674                      DT0026
001636  040326                      DF0026

                    ;               ERROR 32                ;MR1 ERROR READING DATA SYNC
001640  044672                      EM5032
001642  043034                      E5004B
001644  037562                      DT0010
001646  040156                      DF0010
```

```
                    ;       ERROR 33            ;CS1 ERROR AFTER READING SYNC
001650  044672              EM5032
001652  043015              E5004A
001654  037600              DT0011
001656  040202              DF0011

                    ;       ERROR 34            ;MR1 IN ERROR READING DATA OR ECC
001660  044750              EM5034
001662  043034              E5004B
001664  037562              DT0010
001666  040156              DF0010

                    ;       ERROR 35            ;ECC ERROR READING DATA
001670  045015              EM5035
001672  043530              E5021A
001674  037662              DT0021
001676  040302              DF0021

                    ;       ERROR 36            ;CS1 ERROR AFTER READING DATA OR ECC
001700  044750              EM5034
001702  043015              E5004A
001704  037600              DT0011
001706  040202              DF0011

                    ;       ERROR 37            ;DATA COMPARE ERROR (1ST MESSAGE)
001710  045056              EM5037
001712  041232              DH000A
001714  037716              DT0037
001716  040362              DF0037

                    ;       ERROR 40            ;DATA COMPARE ERROR (2ND THRU 10TH ERR)
001720  000000              0
001722  000000              0
001724  037722              DT0040
001726  040402              DF0040

                    ;       ERROR 41
001730  045132              EM5041              ;ECC REGISTER INCORRECT AFTER ECC READ
001732  041232              DH000A
001734  037730              DT0041
001736  040406              DF0041

                    ;       ERROR 42
001740  045217              EM5042              ;ERROR IN ECC PAT CALCULATION AFTER ECC ERROR
001742  041232              DH000A
001744  037662              DT0021
001746  040426              DF0042

                    ;       ERROR 43
001750  045300              EM5043              ;ERROR IN ECC POSITION COUNTING AFTER ECC ERROR
001752  041232              DH000A
001754  037746              DT0043
001756  040446              DF0043

                    ;       ERROR 44
001760  045357              EM5044              ;ERROR AFTER PROCESSING DATA CHECK ERR
```

```
001762  043015                      E5004A
001764  037514                      DT0004
001766  040032                      DF0004

                            ;       ERROR 45
001770  045357                      EM5044                  ;ERROR AFTER PROCESSING DATA CHECK ERR
001772  043111                      E5004E
001774  037760                      DT0045
001776  040466                      DF0045

                            ;       ERROR 46
002000  042751                      EM50046                 ;TRANSFER LENGTH ERROR PARTIAL SEC READ
002002  041232                      DH000A
002004  037770                      DT0046
002006  040512                      DF0046

                            ;       ERROR 47
002010  045521                      EM5047                  ;BSE DID NOT PREVENT DA OR DC INCREMENT
002012  043316                      E5011A
002014  037624                      DT0014
002016  040232                      DF0014

                            ;       ERROR 50
002020  045614                      EM5050                  ;BSE DID NOT CAUSE CERR
002022  043015                      E5004A
002024  037514                      DT0004
002026  040532                      DF0050

                            ;       ERROR 51
002030  045667                      EM5051                  ;BSE FAILED TO SET WHEN EXPECTED
002032  043111                      E5004E
002034  037760                      DT0045
002036  040466                      DF0045

                            ;       ERROR 52
002040  045744                      EM5052                  ;DATA XFER ABORT TO SOON
002042  041232                      DH000A
002044  037770                      DT0046
002046  040512                      DF0046

                            ;       ERROR 53
002050  046064                      EM5053                  ;MR1 ERROR IN WRT CHK OR ECC READ AFTER WRT CHK
002052  043034                      E5004B
002054  037562                      DT0010
002056  040156                      DF0010

                            ;       ERROR 54
002060  046153                      EM5054                  ;ECC ERROR IN WRITE CHECK OP
002062  043530                      E5021A
002064  037662                      DT0021
002066  040302                      DF0021

                            ;       ERROR 55
002070  046064                      EM5053                  ;CS1 ERROR AFTER WRT CHK OR ECC READ IN WRT CHK
002072  043015                      E5004A
002074  037600                      DT0011
```

```
002076  040202                     DF0011

                                   ERROR 56
                        ;
002100  044750                     EM5034               ;CS1 ERROR AFTER READ DATA OR ECC
002102  043015                     E5004A
002104  037514                     DT0004
002106  040032                     DF0004

                                   ERROR 57
                        ;
002110  046064                     EM5053               ;CS1 ERROR AFTER WRITE CHECK OR ECC READ
002112  043015                     E5004A
002114  037514                     DT0004
002116  040032                     DF0004

                                   ERROR 60
                        ;
002120  046216                     EM5060               ;WCE FAILED TO SET IN 16 BIT MODE
002122  041232                     DH000A
002124  037716                     DT0037
002126  040556                     DF0060

                                   ERROR 61
                        ;
002130  046275                     EM5061               ;TRANSFER LENGTH PROBLEM - RKBA
002132  043127                     E5004F
002134  037770                     DT0046
002136  040576                     DF0061

                                   ERROR 62
                        ;
002140  046275                     EM5061               ;TRANSFER LENGTH PROBLEM - RKWC
002142  043145                     E5004G
002144  040000                     DT0062
002146  040622                     DF0062

                                   ERROR 63
                        ;
002150  046352                     EM5063               ;WCE FAILED TO SET IN 18 BIT MODE
002152  043145                     E5004G
002154  040010                     DT0063
002156  040646                     DF0063

                        .SBTTL     TEMPORARY STORAGE FOR RK611 CONTROLLER REGISTER
002160  000000          T.CS1:     .WORD     0          ;CONTROL AND STATUS REGISTER 1
002162  000000          T.WC:      .WORD     0          ;WORD COUNT REGISTER
002164  000000          T.BA:      .WORD     0          ;BUS ADDRESS REGISTER
002166  000000          T.DA:      .WORD     0          ;DESIRED TRACK SECTOR REGISTER
002170  000000          T.CS2:     .WORD     0          ;CONTROL AND STATUS REGISTER 2
002172  000000          T.DS:      .WORD     0          ;DRIVE STATUS REGISTER
002174  000000          T.ER:      .WORD     0          ;ERROR REGISTER
002176  000000          T.ASOF:    .WORD     0          ;ATTENTION SUMMARY AND OFFSET REGISTER
002200  000000          T.DCYL:    .WORD     0          ;DESIRED CYLINDER REGISTER
002202  000000          T.DB:      .WORD     0          ;DATA BUFFER
002204  000000          T.MR1:     .WORD     0          ;MAINTENANCE REGISTER 1
002206  000000          T.MR2:     .WORD     0          ;MAINTENANCE REGISTER 2
002210  000000          T.MR3:     .WORD     0          ;MAINTENANCE REGISTER 3
002212  000000          T.ECPS:    .WORD     0          ;ECC POSITION INFORMATION
002214  000000          T.ECPT:    .WORD     0          ;ECC PATTERN INFORMATION
002216  000000          T.SPAR:    .WORD     0          ;SPARE REGISTER
```

```
                                .SBTTL  EXPECTED RK611 CONTROLLER REGISTERS

        002220  000000          E.CS1:  .WORD   0               ;CONTROL AND STATUS REGISTER 1
        002222  000000          E.WC:   .WORD   0               ;WORD COUNT REGISTER
        002224  000000          E.BA:   .WORD   0               ;BUS ADDRESS REGISTER
        002226  000000          E.DA:   .WORD   0               ;DESIRED TRACK SECTOR REGISTER
        002230  000000          E.CS2:  .WORD   0               ;CONTROL AND STATUS REGISTER 2
        002232  000000          E.DS:   .WORD   0               ;DRIVE STATUS REGISTER
        002234  000000          E.ER:   .WORD   0               ;ERROR REGISTER
        002236  000000          E.ASOF: .WORD   0               ;ATTENTION SUMMARY AND OFFSET REGISTER
        002240  000000          E.DCYL: .WORD   0               ;DESIRED CYLINDER REGISTER
        002242  000000          E.DB:   .WORD   0               ;DATA BUFFER
        002244  000000          E.MR1:  .WORD   0               ;MAINTENANCE REGISTER 1
        002246  000000          E.MR2:  .WORD   0               ;MAINTENANCE REGISTER 2
        002250  000000          E.MR3:  .WORD   0               ;MAINTENANCE REGISTER 3
        002252  000000          E.ECPS: .WORD   0               ;ECC POSITION INFORMATION
        002254  000000          E.ECPT: .WORD   0               ;ECC PATTERN INFORMATION
        002256  000000          E.SPAR: .WORD   0               ;SPARE REGISTER

                                .SBTTL  "L" REGISTERS FOR COMMAND START
        002260  000000          L.CS1:  .WORD   0               ;CONTROL AND STATUS 1
        002262  000000          L.WC:   .WORD   0               ;WORD COUNT
        002264  000000          L.BA:   .WORD   0               ;BUFER ADDRESS
        002266                  L.DA:                           ;DESIRED
        002266     000          L.DS:   .BYTE   0               ;SECTOR
        002267     000          L.DT:   .BYTE   0               ;TRACK
        002270  000000          L.CS2:  .WORD   0               ;CONTROL AND STATUS
        002272  000000          L.ASOF: .WORD   0               ;ATTENTION AND OFFSET
        002274  000000          L.DCYL: .WORD   0               ;DESIRED CYLINDER
        002276  000000          L.MR1:  .WORD   0               ;MAINT. REG 1
        002300  000000          L.MR2:  .WORD   0               ;MAINT. REG 2
        002302  000000          L.MR3:  .WORD   0               ;MAINT. REG 3



                                .SBTTL  PROGRAM DEFINED VARIABLES

        002304  000210          RKVEC:  .WORD   210             ;RK611 VECTOR
        002306  000240          RKPRI:  .WORD   PR5             ;RK611 PRIORITY
        002310  000000          TRAPPC: .WORD   0               ;PC FOR MEMORY CHECK ENABLE TRAP
        002312  000000          SRTFLG: .WORD   0               ;START FLAG
                                                                ;  0 = 200
                                                                ;  1 = 214
                                                                ; -1 = 204
        002314  000000          ERRCNT: .WORD   0               ;ERROR COUNT FOR SWITCH 12 ABORT
        002316  000000          P1.BIT: .WORD   0               ;NEXT BIT IN DATA SIMULATION
        002320  000000          PR.BIT: .WORD   0               ;PRESENT BIT IN DATA SIMULATION
        002322  000000          M1.BIT: .WORD   0               ;PREVIOUS BIT IN DATA SIMULATION
        002324  000000          M2.BIT: .WORD   0               ;BIT BEFORE PREVIOUS BIT
        002326  000000          BITCNT: .WORD   0               ;BIT POSITION
        002330  000000          WRDCNT: .WORD   0               ;WORD COUNT FOR NPR TRANSFER
        002332  000000          MEMPAR: .WORD   0               ;MEMORY EMABLE ON FIRST 24K
        002334  000000          BADPAR: .WORD   0               ;BAD PARITY FLAG
        002336  000000          ECCXOR: .WORD   0               ;ECC XOR SWITCH
        002340  000000          ECCHI:  .WORD   0               ;ECC HIGH STORAGE
        002342  000000          ECCLO:  .WORD   0               ;ECC LOW STORAGE
```

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

```
                                        .SBTTL  PROGRAM SETUP

002360  012737  000001  002312  PARM:   MOV     #1,SRTFLG           ;LOAD START FLAG FOR PARMETER START
002366  000406                          BR      START1

002370  012737  177777  002312  RESTRT: MOV     #-1,SRTFLG          ;LOAD START FLAG FOR RESTART
002376  000402                          BR      START1

002400  005037  002312          START:  CLR     SRTFLG              ;CLEAR START FLAG
002404  000005                  START1: RESET                       ;RESET THE WHOLE SYSTEM
002406  105037  001103                  CLRB    $ERFLG              ;CLEAR ERROR FLAG
002412  012706  001100                  MOV     #STACK,SP           ;INITIALIZE STACK POINTER
002416  012746  000340                  MOV     #PR7,-(SP)          ;LOAD STACK TO LOCK OUT ALL INTERRUPTS
002422  012746  002430                  MOV     #1$,-(SP)           ;LOAD START OF PROGRAM
002426  000002                          RTI                         ;LOAD PSW

002430  004737  035602          1$:     JSR     PC,@#$TKINT         ;INIT KEYBOARD
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002434  012706  001100                  MOV     #$CMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
002440  005026                          CLR     (R6)+               ;;CLEAR MEMORY LOCATION
002442  022706  001140                  CMP     #SWR,R6 ;;DONE?
002446  001374                          BNE     .-6                 ;;LOOP BACK IF NO
002450  012706  001100                  MOV     #STACK,SP           ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002454  012737  033160  000020          MOV     #$SCOPE,@#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
002462  012737  000340  000022          MOV     #340,@#IOTVEC+2     ;;LEVEL 7
002470  012737  034060  000030          MOV     #$ERROR,@#EMTVEC    ;;EMT VECTOR FOR ERROR ROUTINE
002476  012737  000340  000032          MOV     #340,@#EMTVEC+2     ;;LEVEL 7
002504  012737  037420  000034          MOV     #$TRAP,@#TRAPVEC    ;;TRAP VECTOR FOR TRAP CALLS
002512  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002520  012737  037264  000024          MOV     #$PWRDN,@#PWRVEC    ;;POWER FAILURE VECTOR
002526  012737  000340  000026          MOV     #340,@#PWRVEC+2     ;;LEVEL 7
002534  013737  023142  023134          MOV     $ENDCT,$EOPCT       ;;SETUP END-OF-PROGRAM COUNTER
002542  005037  001234                  CLR     $TIMES              ;;INITIALIZE NUMBER OF ITERATIONS
002546  005037  001236                  CLR     $ESCAPE             ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002552  112737  000001  001115          MOVB    #1,$ERMAX           ;;ALLOW ONE ERROR PER TEST
002560  012737  002560  001106          MOV     #.,$LPADR           ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002566  012737  002566  001110          MOV     #.,$LPERR           ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002574  013746  000004                  MOV     @#ERRVEC,-(SP)      ;;SAVE ERROR VECTOR
002600  012737  002634  000004          MOV     #64$,@#ERRVEC       ;;SET UP ERROR VECTOR
002606  012737  177570  001140          MOV     #DSWR,SWR           ;;SETUP FOR A HARDWARE SWICH REGISTER
002614  012737  177570  001142          MOV     #DDISP,DISPLAY      ;;AND A HARDWARE DISPLAY REGISTER
002622  022777  177777  176310          CMP     #-1,@SWR            ;;TRY TO REFERENCE HARDWARE SWR
002630  001012                          BNE     66$                 ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
002632  000403                          BR      65$                 ;;BRANCH IF NO TIMEOUT
002634  012716  002642          64$:    MOV     #65$,(SP)           ;;SET UP FOR TRAP RETURN
002640  000002                          RTI
002642  012737  000176  001140  65$:    MOV     #SWREG,SWR          ;;POINT TO SOFTWARE SWR
002650  012737  000174  001142          MOV     #DISPREG,DISPLAY
002656  012637  000004          66$:    MOV     (SP)+,@#ERRVEC      ;;RESTORE ERROR VECTOR

002662  005037  001256                  CLR     $PASS               ;;CLEAR PASS COUNT
```

```
002666  132737  000200  001271          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002674  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002676  012737  001272  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002704                          67$:
002704  005037  002314                  CLR     ERRCNT          ;CLEAR ERROR COUNT FOR SWITCH 12 ABORT
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
002710  005227  177777                  INC     #-1             ;;FIRST TIME?
002714  001056                          BNE     68$             ;;BRANCH IF NO
002716  022737  023276  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
002724  001452                          BEQ     68$             ;;BRANCH IF YES
002726  104401  002774                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
002732  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
002736  001012                          BNE     70$             ;;BRANCH IF YES
002740  123727  001270  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
002746  001406                          BEQ     70$             ;;BRANCH IF YES
002750  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
002756  001005                          BNE     71$             ;;BRANCH IF NO
002760  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
002762  000403                          BR      71$
002764  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
002772                          71$:
002772  000427                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF>/RK611 DISKLESS DIAGNOSTIC: PART 5  CZR6EB0/<CRLF>
003052                          68$:
003052  005227  177777                  INC     #-1             ;TEST IF PASS 0
003056  001002                          BNE     4$              ;NO - SKIP
003060  104401  041004                  TYPE    ,OPR006         ;TYPE PASS TIME MESSAGE
003064  022737  000001  002312  4$:     CMP     #1,SRTFLG       ;CHECK IF PARAMETER START
003072  001121                          BNE     15$             ;NO, CONTINUE SETUP
003074  104401  040672          5$:     TYPE    ,OPR001         ;TYPE "RK611 BUS ADDRESS (   ) ="
003100  013746  001324                  MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
003104  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
003106  104401  040724                  TYPE    ,OPR002
003112  104412                          RDOCT                   ;GET VALUE
003114  012637  001202                  MOV     (SP)+,$TMP0
003120  001407                          BEQ     7$              ;CHECK IF <CR>
003122  022737  160000  001202          CMP     #160000,$TMP0   ;CHECK IF IN I/O PAGE
003130  101361                          BHI     5$
003132  013737  001202  001324          MOV     $TMP0,$BASE     ;LOAD NEW BUS ADDRESS
003140  104401  040732          7$:     TYPE    ,OPR003         ;TYPE "RK611 VECTOR ADDRESS (   ) ="
003144  013746  001320                  MOV     $VECT1,-(SP)    ;TYPE OUT VECTOR ADDRESS
003150  042716  160000                  BIC     #160000,(SP)
003154  104401  040724                  TYPE    ,OPR002
003160  104412                          RDOCT                   ;GET VALUE
003162  012637  001202                  MOV     (SP)+,$TMP0
003166  001412                          BEQ     10$             ;CHECK IF <CR>
003170  022737  001000  001202          CMP     #1000,$TMP0     ;CHECK IF LEGAL
003176  101760                          BLOS    7$
003200  042737  017777  001320          BIC     #17777,$VECT1   ;LOAD NEW VECTOR ADDRESS
003206  053737  001202  001320          BIS     $TMP0,$VECT1
003214  104401  040762          10$:    TYPE    ,OPR004         ;TYPE "RK611 PRIORITY (   ) ="
003220  005046                          CLR     -(SP)
003222  113716  001321                  MOVB    $VECT1+1,(SP)
003226  006216                          ASR     (SP)            ;SHIFT 5 BITS RIGHT
```

```
003230  006216                                   ASR     (SP)
003232  006216                                   ASR     (SP)
003234  006216                                   ASR     (SP)
003236  006216                                   ASR     (SP)
003240  104402                                   TYPOC
003242  104401  040724                           TYPE    ,OPR002
003246  104412                                   RDOCT                    ;GET VALUE
003250  012637  001202                           MOV     (SP)+,$TMP0
003254  001430                                   BEQ     15$              ;CHECK FOR DEFAULT
003256  022737  000007  001202                   CMP     #7,$TMP0         ;CHECK IF LEGAL
003264  103753                                   BLO     10$
003266  022737  000004  001202                   CMP     #4,$TMP0
003274  101347                                   BHI     10$
003276  006337  001202                           ASL     $TMP0            ;SHIFT 5 BITS LEFT
003302  006337  001202                           ASL     $TMP0
003306  006337  001202                           ASL     $TMP0
003312  006337  001202                           ASL     $TMP0
003316  006337  001202                           ASL     $TMP0
003322  042737  160000  001320                   BIC     #160000,$VECT1   ;STORE NEW PRIORITY
003330  153737  001202  001321                   BISB    $TMP0,$VECT1+1
003336  013737  001320  002304  15$:             MOV     $VECT1,RKVEC     ;STORE RK611 VECTOR
003344  042737  160000  002304                   BIC     #160000,RKVEC
003352  113737  001321  002306                   MOVB    $VECT1+1,RKPRI   ;STORE PRIORITY
003360  004737  032652                           JSR     PC,$SIZE         ;SIZE MEMORY

003364  013702  001324          NEWPAS:          MOV     $BASE,R2         ;SET RK POINTER
003370  005037  001236                           CLR     $ESCAPE          ;CLEAR ESCAPE
003374  004737  032516                           JSR     PC,PARCHK        ;CHECK OF MEMORY CHECK ENABLE
003400  012746  000000                           MOV     #PR0,-(SP)       ;LOCK OUT INTERRUPTS
003404  012746  003412                           MOV     #TST1,-(SP)
003410  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.
                                        ;*
                                        ;;**************************************************************
003412  000004                          TST1:   SCOPE
003414  012737  000012  001234                  MOV     #10.,$TIMES     ;;DO 10. ITERATIONS

003422  012762  100000  000000                  MOV     #CCLR,RKCS1(R2) ;CLR CONTROLLER

003430  005037  002352                          CLR     BSEDES          ;CLEAR BAD SEC DESIRED
003434  004437  023316                          JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE TO SET
                                                                        ;ECCW IN RKMR1 FROM INIT POWER UP

003440  004437  032230                          JSR     R4,LOADRK       ;LOAD "L" REGISTERS
003444  000000                                  0                       ;CYLINDER 0
003446     000                                  .BYTE   0               ;SECTOR 0
003447     000                                  .BYTE   0               ;TRACK 0
003450  047636                                  OBUFF                   ;BUFFER ADDRESS OBUFF
003452  177377                                  -401                    ;WORD COUNT -401
003454  000023                                  WRDATA                  ;COMMAND WRDATA

003456  004437  032014                          JSR     R4,BLDDAT       ;GO BUILD DATA
003462  000001                                  1                       ;PATTERN 1

003464  004437  032274                          JSR     R4,OPSTRT       ;START THE OPERATION

003470  004437  023334                          JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
003474  104004                                  ERROR   4               ;CS1 MISCOMPARE
003476  104005                                  ERROR   5               ;MR1     "
003500  104006                                  ERROR   6               ;MR2     "
003502  104007                                  ERROR   7               ;MR3     "

003504  004437  023316                          JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

003510  004437  023740                          JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
003514  104010                                  ERROR   10              ;MR1 CONTENTS IN ERROR
003516  104011                                  ERROR   11              ;CS1 CONTENTS IN ERROR

003520  004437  024276                          JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
003524  104012                                  ERROR   12              ;MR1 CONTENTS IN ERROR
003526  104013                                  ERROR   13              ;CS1 IN ERROR AFTER SEARCH
003530  104014                                  ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
```

```
        003532  012737  043403  001510          MOV     #EM5020,EMW             ;SET MESSAGE FOR WRITE FAILURE
        003540  004437  025204                  JSR     R4,DWGPSN               ;SIMULATE GAP AND SYNC FOR WRITE
        003544  104015                          ERROR   15                      ;MR1 IN ERROR IN READ GAP
        003546  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READ GAP
        003550  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING SYNC
        003552  104020                          ERROR   20                      ;CS1 IN ERROR AFTER WRITING SYNC

        003554  012737  043462  001510          MOV     #EM5021,EMW             ;SET MESSAGE FOR WRITE FAILURE
        003562  004437  025604                  JSR     R4,DWRITE               ;SIMULATE WRITE DATA AND ECC
        003566  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING DATA OR ECC
        003570  104021                          ERROR   21                      ;ECC IN ERROR WHILE WRITING DATA
        003572  104022                          ERROR   22                      ;CS1 IN ERROR AFTER WRITING DATA AND ECC

        003574  012700  000034                  MOV     #7*4,R0                 ;ISSUE CLOCKS TO NEXT HDR COMPARE
        003600  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)             ;CLOCK LOOP
        003606  012762  000040  000026          MOV     #DMD,RKMR1(R2)
        003614  005300                          DEC     R0
        003616  001370                          BNE     1$

        003620  013703  002274                  MOV     L.DCYL,R3               ;GET DESIRED CYLINDER
        003624  004437  032444                  JSR     R4,FSBLVV               ;INVERT WORD
        003630  010337  002246                  MOV     R3,E.MR2                ;STORE EXPECTED MR2
        003634  113703  002267                  MOVB    L.DT,R3                 ;GET DESIRED TRACK/SECTOR
        003640  042703  177400                  BIC     #177400,R3              ;CLEAR UNUSED BITS
        003644  012700  000005                  MOV     #5,R0                   ;SET COUNT FOR SHIFTING
        003650  006303                  5$:     ASL     R3                      ;SHIFT TRACK FOR HEADER ALIGNMENT
        003652  005300                          DEC     R0                      ;DEC COUNT
        003654  001375                          BNE     5$                      ;LOOP UNTIL ZERO
        003656  153703  002266                  BISB    L.DS,R3                 ;INSERT SECTOR  NUMBER

        003662  032737  010000  002260          BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
        003670  001402                          BEQ     6$                      ;NO - SKIP
        003672  052703  001000                  BIS     #BIT9,R3                ;SET FORMAT BIT FOR WORD TWO
        003676                          6$:
        003676  004437  032444                  JSR     R4,FSBLVV               ;INVERT WORD
        003702  010337  002250                  MOV     R3,E.MR3                ;STORE EXPECTED MR3
        003706  016237  000034  002206          MOV     RKMR2(R2),T.MR2         ;GET MR2
        003714  016237  000036  002210          MOV     RKMR3(R2),T.MR3         ;GET MR3
        003722  016237  000000  002160          MOV     RKCS1(R2),T.CS1         ;GET CS1
        003730  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
        003736  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF OK
        003744  001402                          BEQ     3$                      ;YES - SKIP
        003746  104025                          ERROR   25                      ;CS1 IN ERROR AT MULTI-SECTOR TIME
        003750  000427                          BR      TST2                    ;;GO TO NEXT TEST

        003752  023737  002206  002246  3$:     CMP     T.MR2,E.MR2             ;CHECK IF MR2 OK
        003760  001402                          BEQ     2$                      ;YES - SKIP
        003762  104023                          ERROR   23                      ;MR2 IN ERROR AT MULTI-SECTOR TIME
        003764  000421                          BR      TST2                    ;;GO TO NEXT TEST

        003766  023737  002210  002250  2$:     CMP     T.MR3,E.MR3             ;CHECK IF MR3 OK
        003774  001402                          BEQ     4$                      ;YES - SKIP
        003776  104024                          ERROR   24                      ;MR3 IN ERROR AT MULTI-SECTOR TIME
        004000  000413                          BR      TST2                    ;;GO TO NEXT TEST

        004002  004437  023316          4$:     JSR     R4,DSECTR               ;SIMULATE 2ND SECTOR PULSE
```

```

004006  004437  023740                  JSR     R4,DRSYNC          ;SIMULATE PREAMBLE
004012  104010                          ERROR   10                 ;MR1 ERROR
004014  104011                          ERROR   11                 ;CS1 ERROR

004016  004437  024276                  JSR     R4,DHDCMP          ;SIMULATE HEADER SEARCH
004022  104012                          ERROR   12                 ;MR1 ERROR
004024  104013                          ERROR   13                 ;CS1 ERROR
004026  104014                          ERROR   14                 ;RKDCYL OR RKDA IN ERROR

                                ;;*****************************************************************
                                ;*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.
                                ;*
                                ;;*****************************************************************
004030  000004                  TST2:   SCOPE
004032  012737  000012  001234          MOV     #10.,$TIMES        ;;DO 10. ITERATIONS

004040  012762  100000  000000          MOV     #CCLR,RKCS1(R2)    ;CLEAR CONTROLLER

004046  005037  002352                  CLR     BSEDES             ;CLEAR BAD SEC DESIRED

004052  004437  032230                  JSR     R4,LOADRK          ;LOAD "L" REGISTERS
004056  000000                          0                          ;CYLINDER 0
004060     023                          .BYTE   23                 ;SECTOR 23
004061     000                          .BYTE   0                  ;TRACK 0
004062  047636                          OBUFF                      ;BUFFER ADDRESS OBUFF
004064  177377                          -401                       ;WORD COUNT -401
004066  000023                          WRDATA                     ;COMMAND WRDATA

004070  004437  032014                  JSR     R4,BLDDAT          ;GO BUILD DATA
004074  000002                          2                          ;PATTERN 2

004076  004437  032274                  JSR     R4,OPSTRT          ;START THE OPERATION

004102  004437  023334                  JSR     R4,DISEEK          ;SIMULATE IMPLIED SEEK
004106  104004                          ERROR   4                  ;CS1 MISCOMPARE
004110  104005                          ERROR   5                  ;MR1     "
004112  104006                          ERROR   6                  ;MR2     "
004114  104007                          ERROR   7                  ;MR3     "

004116  004437  023316                  JSR     R4,DSECTR          ;SIMULATE SECTOR PULSE

004122  004437  023740                  JSR     R4,DRSYNC          ;SIMULATE HEADER PREAMBLE
004126  104010                          ERROR   10                 ;MR1 CONTENTS IN ERROR
004130  104011                          ERROR   11                 ;CS1 CONTENTS IN ERROR
```

```

004132  004437  024276                  JSR     R4,DHDCMP          ;SIMULATE HEADER SEARCH
004136  104012                          ERROR   12                 ;MR1 CONTENTS IN ERROR
004140  104013                          ERROR   13                 ;CS1 IN ERROR AFTER SEARCH
004142  104014                          ERROR   14                 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

004144  012737  043403  001510          MOV     #EMS020,EMW        ;SET MESSAGE FOR WRITE FAILURE
004152  004437  025204                  JSR     R4,DWGPSN          ;SIMULATE GAP AND SYNC FOR WRITE
004156  104015                          ERROR   15                 ;MR1 IN ERROR IN READ GAP
004160  104016                          ERROR   16                 ;CS1 IN ERROR AFTER READ GAP
004162  104017                          ERROR   17                 ;MR1 IN ERROR IN WRITING SYNC
004164  104020                          ERROR   20                 ;CS1 IN ERROR AFTER WRITING SYNC

004166  012737  043462  001510          MOV     #EMS021,EMW        ;SET MESSAGE FOR WRITE FAILURE
004174  004437  025604                  JSR     R4,DWRITE          ;SIMULATE WRITE DATA AND ECC
004200  104017                          ERROR   17                 ;MR1 IN ERROR IN WRITING DATA OR ECC
004202  104021                          ERROR   21                 ;ECC IN ERROR WHILE WRITING DATA
004204  104022                          ERROR   22                 ;CS1 IN ERROR AFTER WRITING DATA AND ECC

004206  012700  000034                  MOV     #7*4,R0            ;ISSUE CLOCKS TO NEXT HDR COMPARE
004212  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)        ;CLOCK LOOP
004220  012762  000040  000026          MOV     #DMD,RKMR1(R2)
004226  005300                          DEC     R0
004230  001370                          BNE     1$

004232  013703  002274                  MOV     L.DCYL,R3          ;GET DESIRED CYLINDER
004236  004437  032444                  JSR     R4,FSBLVV          ;INVERT WORD
004242  010337  002246                  MOV     R3,E.MR2           ;STORE EXPECTED MR2
004246  113703  002267                  MOVB    L.DT,R3            ;GET DESIRED TRACK/SECTOR
004252  042703  177400                  BIC     #177400,R3         ;CLEAR UNUSED BITS
004256  012700  000005                  MOV     #5,R0              ;SET COUNT FOR SHIFTING
004262  006303                  5$:     ASL     R3                 ;SHIFT TRACK FOR HEADER ALIGNMENT
004264  005300                          DEC     R0                 ;DEC COUNT
004266  001375                          BNE     5$                 ;LOOP UNTIL ZERO
004270  153703  002266                  BISB    L.DS,R3            ;INSERT SECTOR  NUMBER

004274  032737  010000  002260          BIT     #CFMT,L.CS1        ;TEST IF 18 BIT MODE
004302  001402                          BEQ     6$                 ;NO - SKIP
004304  052703  001000                  BIS     #BIT9,R3           ;SET FORMAT BIT FOR WORD TWO
004310                          6$:
004310  004437  032444                  JSR     R4,FSBLVV          ;INVERT WORD
004314  010337  002250                  MOV     R3,E.MR3           ;STORE EXPECTED MR3
004320  016237  000034  002206          MOV     RKMR2(R2),T.MR2    ;GET MR2
004326  016237  000036  002210          MOV     RKMR3(R2),T.MR3    ;GET MR3
004334  016237  000000  002160          MOV     RKCS1(R2),T.CS1    ;GET CS1
004342  013737  002260  002220          MOV     L.CS1,E.CS1        ;GET EXPECTED CS1
004350  023737  002160  002220          CMP     T.CS1,E.CS1        ;CHECK IF OK
004356  001402                          BEQ     3$                 ;YES - SKIP
004360  104025                          ERROR   25                 ;CS1 IN ERROR AT MULTI-SECTOR TIME
004362  000427                          BR      TST3               ;;GO TO NEXT TEST

004364  023737  002206  002246  3$:     CMP     T.MR2,E.MR2        ;CHECK IF MR2 OK
004372  001402                          BEQ     2$                 ;YES - SKIP
004374  104023                          ERROR   23                 ;MR2 IN ERROR AT MULTI-SECTOR TIME
004376  000421                          BR      TST3               ;;GO TO NEXT TEST

```

```
004400  023737  002210  002250  2$:     CMP     T.MR3,E.MR3          ;CHECK IF MR3 OK
004406  001402                          BEQ     4$                   ;YES - SKIP
004410  104024                          ERROR   24                   ;MR3 IN ERROR AT MULTI-SECTOR TIME
004412  000413                          BR      TST3                 ;;GO TO NEXT TEST

004414  004437  023316          4$:     JSR     R4,DSECTR            ;SIMULATE 2ND SECTOR PULSE

004420  004437  023740                  JSR     R4,DRSYNC            ;SIMULATE PREAMBLE
004424  104010                          ERROR   10                   ;MR1 ERROR
004426  104011                          ERROR   11                   ;CS1 ERROR

004430  004437  024276                  JSR     R4,DHDCMP            ;SIMULATE HEADER SEARCH
004434  104012                          ERROR   12                   ;MR1 ERROR
004436  104013                          ERROR   13                   ;CS1 ERROR
004440  104014                          ERROR   14                   ;RKDCYL OR RKDA IN ERROR

                                ;;*************************************************************
                                ;*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.
                                ;*
                                ;;*************************************************************
004442  000004                  TST3:   SCOPE
004444  012737  000012  001234          MOV     #10.,$TIMES          ;;DO 10. ITERATIONS

004452  012762  100000  000000          MOV     #CCLR,RKCS1(R2)      ;CLEAR CONTROLLER

004460  005037  002352                  CLR     BSEDES               ;CLEAR BAD SEC DESIRED

004464  004437  032230                  JSR     R4,LOADRK            ;LOAD "L" REGISTERS
004470  000000                          0                            ;CYLINDER 0
004472     025                          .BYTE   25                   ;SECTOR 25
004473     000                          .BYTE   0                    ;TRACK 0
004474  047636                          OBUFF                        ;BUFFER ADDRESS OBUFF
004476  177377                          -401                         ;WORD COUNT -401
004500  000023                          WRDATA                       ;COMMAND WRDATA

004502  004437  032014                  JSR     R4,BLDDAT            ;GO BUILD DATA
004506  000003                          3                            ;PATTERN 3

004510  004437  032274                  JSR     R4,OPSTRT            ;START THE OPERATION

004514  004437  023334                  JSR     R4,DISEEK            ;SIMULATE IMPLIED SEEK
004520  104004                          ERROR   4                    ;CS1 MISCOMPARE
004522  104005                          ERROR   5                    ;MR1     "
004524  104006                          ERROR   6                    ;MR2     "
004526  104007                          ERROR   7                    ;MR3     "
```

```
004530  004437  023316                    JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

004534  004437  023740                    JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
004540  104010                            ERROR   10                  ;MR1 CONTENTS IN ERROR
004542  104011                            ERROR   11                  ;CS1 CONTENTS IN ERROR

004544  004437  024276                    JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
004550  104012                            ERROR   12                  ;MR1 CONTENTS IN ERROR
004552  104013                            ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
004554  104014                            ERROR   14                  ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

004556  012737  043403  001510            MOV     #EM5020,EMW         ;SET MESSAGE FOR WRITE FAILURE
004564  004437  025204                    JSR     R4,DWGPSN           ;SIMULATE GAP AND SYNC FOR WRITE
004570  104015                            ERROR   15                  ;MR1 IN ERROR IN READ GAP
004572  104016                            ERROR   16                  ;CS1 IN ERROR AFTER READ GAP
004574  104017                            ERROR   17                  ;MR1 IN ERROR IN WRITING SYNC
004576  104020                            ERROR   20                  ;CS1 IN ERROR AFTER WRITING SYNC

004600  012737  043462  001510            MOV     #EM5021,EMW         ;SET MESSAGE FOR WRITE FAILURE
004606  004437  025604                    JSR     R4,DWRITE           ;SIMULATE WRITE DATA AND ECC
004612  104017                            ERROR   17                  ;MR1 IN ERROR IN WRITING DATA OR ECC
004614  104021                            ERROR   21                  ;ECC IN ERROR WHILE WRITING DATA
004616  104022                            ERROR   22                  ;CS1 IN ERROR AFTER WRITING DATA AND ECC

004620  012700  000324                    MOV     #53.*4,R0                   ;ISSUE CLOCKS TO NEXT HDR COMPARE
004624  012762  000440  000026    1$:     MOV     #DMD!MCLK,RKMR1(R2)         ;CLOCK LOOP
004632  012762  000040  000026            MOV     #DMD,RKMR1(R2)
004640  005300                            DEC     R0
004642  001370                            BNE     1$

004644  013703  002274                    MOV     L.DCYL,R3           ;GET DESIRED CYLINDER
004650  004437  032444                    JSR     R4,FSBLVV           ;INVERT WORD
004654  010337  002246                    MOV     R3,E.MR2            ;STORE EXPECTED MR2
004660  113703  002267                    MOVB    L.DT,R3             ;GET DESIRED TRACK/SECTOR
004664  042703  177400                    BIC     #177400,R3          ;CLEAR UNUSED BITS
004670  012700  000005                    MOV     #5,R0               ;SET COUNT FOR SHIFTING
004674  006303                    5$:     ASL     R3                  ;SHIFT TRACK FOR HEADER ALIGNMENT
004676  005300                            DEC     R0                  ;DEC COUNT
004700  001375                            BNE     5$                  ;LOOP UNTIL ZERO
004702  153703  002266                    BISB    L.DS,R3             ;INSERT SECTOR  NUMBER

004706  032737  010000  002260            BIT     #CFMT,L.CS1         ;TEST IF 18 BIT MODE
004714  001402                            BEQ     6$                  ;NO - SKIP
004716  052703  001000                    BIS     #BIT9,R3            ;SET FORMAT BIT FOR WORD TWO
004722                            6$:
004722  004437  032444                    JSR     R4,FSBLVV           ;INVERT WORD
004726  010337  002250                    MOV     R3,E.MR3            ;STORE EXPECTED MR3
004732  016237  000034  002206            MOV     RKMR2(R2),T.MR2     ;GET MR2
004740  016237  000036  002210            MOV     RKMR3(R2),T.MR3     ;GET MR3
004746  016237  000000  002160            MOV     RKCS1(R2),T.CS1     ;GET CS1
004754  013737  002260  002220            MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
004762  023737  002160  002220            CMP     T.CS1,E.CS1         ;CHECK IF OK
004770  001402                            BEQ     3$                  ;YES - SKIP
004772  104025                            ERROR   25                  ;CS1 IN ERROR AT MULTI-SECTOR TIME
004774  000427                            BR      TST4                ;;GO TO NEXT TEST
```

```
004776  023737  002206  002246  3$:     CMP     T.MR2,E.MR2         ;CHECK IF MR2 OK
005004  001402                          BEQ     2$                  ;YES - SKIP
005006  104023                          ERROR   23                  ;MR2 IN ERROR AT MULTI-SECTOR TIME
005010  000421                          BR      TST4                ;;GO TO NEXT TEST

005012  023737  002210  002250  2$:     CMP     T.MR3,E.MR3         ;CHECK IF MR3 OK
005020  001402                          BEQ     4$                  ;YES - SKIP
005022  104024                          ERROR   24                  ;MR3 IN ERROR AT MULTI-SECTOR TIME
005024  000413                          BR      TST4                ;;GO TO NEXT TEST

005026  004437  023316          4$:     JSR     R4,DSECTR           ;SIMULATE 2ND SECTOR PULSE

005032  004437  023740                  JSR     R4,DRSYNC           ;SIMULATE PREAMBLE
005036  104010                          ERROR   10                  ;MR1 ERROR
005040  104011                          ERROR   11                  ;CS1 ERROR

005042  004437  024276                  JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
005046  104012                          ERROR   12                  ;MR1 ERROR
005050  104013                          ERROR   13                  ;CS1 ERROR
005052  104014                          ERROR   14                  ;RKDCYL OR RKDA IN ERROR

                                ;;*****************************************************************
                                ;*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.
                                ;*
                                ;;*****************************************************************
005054  000004                  TST4:   SCOPE
005056  012737  000012  001234          MOV     #10.,$TIMES         ;;DO 10. ITERATIONS

005064  012762  100000  000000          MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER

005072  005037  002352                  CLR     BSEDES              ;CLEAR BAD SEC DESIRED

005076  004437  032230                  JSR     R4,LOADRK           ;LOAD "L" REGISTERS
005102  000000                          0                           ;CYLINDER 0
005104     025                          .BYTE   25                  ;SECTOR 25
005105     002                          .BYTE   2                   ;TRACK 2
005106  047636                          OBUFF                       ;BUFFER ADDRESS OBUFF
005110  177377                          -401                        ;WORD COUNT -401
005112  000023                          WRDATA                      ;COMMAND WRDATA

005114  004437  032014                  JSR     R4,BLDDAT           ;GO BUILD DATA
005120  000004                          4                           ;PATTERN 4

005122  004437  032274                  JSR     R4,OPSTRT           ;START THE OPERATION

005126  004437  023334                  JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
```

```
005132  104004                          ERROR   4                       ;CS1 MISCOMPARE
005134  104005                          ERROR   5                       ;MR1     "
005136  104006                          ERROR   6                       ;MR2     "
005140  104007                          ERROR   7                       ;MR3     "

005142  004437  023316                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

005146  004437  023740                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
005152  104010                          ERROR   10                      ;MR1 CONTENTS IN ERROR
005154  104011                          ERROR   11                      ;CS1 CONTENTS IN ERROR

005156  004437  024276                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
005162  104012                          ERROR   12                      ;MR1 CONTENTS IN ERROR
005164  104013                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
005166  104014                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

005170  012737  043403  001510          MOV     #EM5020,EMW             ;SET MESSAGE FOR WRITE FAILURE
005176  004437  025204                  JSR     R4,DWGPSN               ;SIMULATE GAP AND SYNC FOR WRITE
005202  104015                          ERROR   15                      ;MR1 IN ERROR IN READ GAP
005204  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READ GAP
005206  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING SYNC
005210  104020                          ERROR   20                      ;CS1 IN ERROR AFTER WRITING SYNC

005212  012737  043462  001510          MOV     #EM5021,EMW             ;SET MESSAGE FOR WRITE FAILURE
005220  004437  025604                  JSR     R4,DWRITE               ;SIMULATE WRITE DATA AND ECC
005224  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING DATA OR ECC
005226  104021                          ERROR   21                      ;ECC IN ERROR WHILE WRITING DATA
005230  104022                          ERROR   22                      ;CS1 IN ERROR AFTER WRITING DATA AND ECC

005232  012700  000324                  MOV     #53.*4,R0                       ;ISSUE CLOCKS TO NEXT HDR COMPARE
005236  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)             ;CLOCK LOOP
005244  012762  000040  000026          MOV     #DMD,RKMR1(R2)
005252  005300                          DEC     R0
005254  001370                          BNE     1$

005256  013703  002274                  MOV     L.DCYL,R3               ;GET DESIRED CYLINDER
005262  004437  032444                  JSR     R4,FSBLVV               ;INVERT WORD
005266  010337  002246                  MOV     R3,E.MR2                ;STORE EXPECTED MR2
005272  113703  002267                  MOVB    L.DT,R3                 ;GET DESIRED TRACK/SECTOR
005276  042703  177400                  BIC     #177400,R3              ;CLEAR UNUSED BITS
005302  012700  000005                  MOV     #5,R0                   ;SET COUNT FOR SHIFTING
005306  006303                  5$:     ASL     R3                      ;SHIFT TRACK FOR HEADER ALIGNMENT
005310  005300                          DEC     R0                      ;DEC COUNT
005312  001375                          BNE     5$                      ;LOOP UNTIL ZERO
005314  153703  002266                  BISB    L.DS,R3                 ;INSERT SECTOR  NUMBER

005320  032737  010000  002260          BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
005326  001402                          BEQ     6$                      ;NO - SKIP
005330  052703  001000                  BIS     #BIT9,R3                ;SET FORMAT BIT FOR WORD TWO
005334                          6$:
005334  004437  032444                  JSR     R4,FSBLVV               ;INVERT WORD
005340  010337  002250                  MOV     R3,E.MR3                ;STORE EXPECTED MR3
005344  016237  000034  002206          MOV     RKMR2(R2),T.MR2         ;GET MR2
005352  016237  000036  002210          MOV     RKMR3(R2),T.MR3         ;GET MR3
005360  016237  000000  002160          MOV     RKCS1(R2),T.CS1         ;GET CS1
005366  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
```

```
005374  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK IF OK
005402  001402                          BEQ     3$                  ;YES - SKIP
005404  104025                          ERROR   25                  ;CS1 IN ERROR AT MULTI-SECTOR TIME
005406  000427                          BR      TST5                ;;GO TO NEXT TEST

005410  023737  002206  002246  3$:     CMP     T.MR2,E.MR2         ;CHECK IF MR2 OK
005416  001402                          BEQ     2$                  ;YES - SKIP
005420  104023                          ERROR   23                  ;MR2 IN ERROR AT MULTI-SECTOR TIME
005422  000421                          BR      TST5                ;;GO TO NEXT TEST

005424  023737  002210  002250  2$:     CMP     T.MR3,E.MR3         ;CHECK IF MR3 OK
005432  001402                          BEQ     4$                  ;YES - SKIP
005434  104024                          ERROR   24                  ;MR3 IN ERROR AT MULTI-SECTOR TIME
005436  000413                          BR      TST5                ;;GO TO NEXT TEST

005440  004437  023316          4$:     JSR     R4,DSECTR           ;SIMULATE 2ND SECTOR PULSE

005444  004437  023740                  JSR     R4,DRSYNC           ;SIMULATE PREAMBLE
005450  104010                          ERROR   10                  ;MR1 ERROR
005452  104011                          ERROR   11                  ;CS1 ERROR

005454  004437  024276                  JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
005460  104012                          ERROR   12                  ;MR1 ERROR
005462  104013                          ERROR   13                  ;CS1 ERROR
005464  104014                          ERROR   14                  ;RKDCYL OR RKDA IN ERROR


                                ;;*************************************************************
                                ;*TEST 5        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 RECOGNITION
                                ;*      OCCURS.
                                ;*
                                ;;*************************************************************
005466  000004                  TST5:   SCOPE
005470  012737  000012  001234          MOV     #10.,$TIMES         ;;DO 10. ITERATIONS

005476  012762  100000  000000          MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER

005504  005037  002352                  CLR     BSEDES              ;CLEAR BAD SEC DESIRED
005510  012700  002000                  MOV     #2000,R0            ;SET A COUNT TO DELAY FOR CLEAR
005514  005300                  20$:    DEC     R0
005516  001376                          BNE     20$                 ;LOOP UNTIL ZERO

005520  004437  032230                  JSR     R4,LOADRK           ;LOAD "L" REGISTERS
005524  000000                          0                           ;CYLINDER 0
005526     023                          .BYTE   23                  ;SECTOR 23
005527     000                          .BYTE   0                   ;TRACK 0
005530  047636                          OBUFF                       ;BUFFER ADDRESS OBUFF
```

```
005532  177377                          -401                            ;WORD COUNT -401
005534  010023                          WRDATA!CFMT                             ;COMMAND WRDATA!CFMT

005536  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
005542  000007                          7                               ;PATTERN 7

005544  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION

005550  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
005554  104004                          ERROR   4                       ;CS1 MISCOMPARE
005556  104005                          ERROR   5                       ;MR1      "
005560  104006                          ERROR   6                       ;MR2      "
005562  104007                          ERROR   7                       ;MR3      "

005564  004437  023316                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

005570  004437  023740                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
005574  104010                          ERROR   10                      ;MR1 CONTENTS IN ERROR
005576  104011                          ERROR   11                      ;CS1 CONTENTS IN ERROR

005600  004437  024276                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
005604  104012                          ERROR   12                      ;MR1 CONTENTS IN ERROR
005606  104013                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
005610  104014                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

005612  012737  043403  001510          MOV     #EM5020,EMW             ;SET MESSAGE FOR WRITE FAILURE
005620  004437  025204                  JSR     R4,DWGPSN               ;SIMULATE GAP AND SYNC FOR WRITE
005624  104015                          ERROR   15                      ;MR1 IN ERROR IN READ GAP
005626  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READ GAP
005630  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING SYNC
005632  104020                          ERROR   20                      ;CS1 IN ERROR AFTER WRITING SYNC

005634  012737  043462  001510          MOV     #EM5021,EMW             ;SET MESSAGE FOR WRITE FAILURE
005642  004437  025604                  JSR     R4,DWRITE               ;SIMULATE WRITE DATA AND ECC
005646  104017                          ERROR   17                      ;MR1 IN ERROR IN WRITING DATA OR ECC
005650  104021                          ERROR   21                      ;ECC IN ERROR WHILE WRITING DATA
005652  104022                          ERROR   22                      ;CS1 IN ERROR AFTER WRITING DATA AND ECC

005654  012700  000324                  MOV     #53.*4,R0                       ;ISSUE CLOCKS TO NEXT HDR COMPARE
005660  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)             ;CLOCK LOOP
005666  012762  000040  000026          MOV     #DMD,RKMR1(R2)
005674  005300                          DEC     R0
005676  001370                          BNE     1$

005700  013703  002274                  MOV     L.DCYL,R3               ;GET DESIRED CYLINDER
005704  004437  032444                  JSR     R4,FSBLVV               ;INVERT WORD
005710  010337  002246                  MOV     R3,E.MR2                ;STORE EXPECTED MR2
005714  113703  002267                  MOVB    L.DT,R3                 ;GET DESIRED TRACK/SECTOR
005720  042703  177400                  BIC     #177400,R3              ;CLEAR UNUSED BITS
005724  012700  000005                  MOV     #5,R0                   ;SET COUNT FOR SHIFTING
005730  006303                  5$:     ASL     R3                      ;SHIFT TRACK FOR HEADER ALIGNMENT
005732  005300                          DEC     R0                      ;DEC COUNT
005734  001375                          BNE     5$                      ;LOOP UNTIL ZERO
005736  153703  002266                  BISB    L.DS,R3                 ;INSERT SECTOR  NUMBER

005742  032737  010000  002260          BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
```

```
005750  001402                                   BEQ     6$                      ;NO - SKIP
005752  052703  001000                           BIS     #BIT9,R3                ;SET FORMAT BIT FOR WORD TWO
005756                                   6$:
005756  004437  032444                           JSR     R4,FSBLVV               ;INVERT WORD
005762  010337  002250                           MOV     R3,E.MR3                ;STORE EXPECTED MR3
005766  016237  000034  002206                   MOV     RKMR2(R2),T.MR2         ;GET MR2
005774  016237  000036  002210                   MOV     RKMR3(R2),T.MR3         ;GET MR3
006002  016237  000000  002160                   MOV     RKCS1(R2),T.CS1         ;GET CS1
006010  013737  002260  002220                   MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
006016  023737  002160  002220                   CMP     T.CS1,E.CS1             ;CHECK IF OK
006024  001402                                   BEQ     3$                      ;YES - SKIP
006026  104025                                   ERROR   25                      ;CS1 IN ERROR AT MULTI-SECTOR TIME
006030  000427                                   BR      TST6                    ;;GO TO NEXT TEST

006032  023737  002206  002246   3$:             CMP     T.MR2,E.MR2             ;CHECK IF MR2 OK
006040  001402                                   BEQ     2$                      ;YES - SKIP
006042  104023                                   ERROR   23                      ;MR2 IN ERROR AT MULTI-SECTOR TIME
006044  000421                                   BR      TST6                    ;;GO TO NEXT TEST

006046  023737  002210  002250   2$:             CMP     T.MR3,E.MR3             ;CHECK IF MR3 OK
006054  001402                                   BEQ     4$                      ;YES - SKIP
006056  104024                                   ERROR   24                      ;MR3 IN ERROR AT MULTI-SECTOR TIME
006060  000413                                   BR      TST6                    ;;GO TO NEXT TEST

006062  004437  023316           4$:             JSR     R4,DSECTR               ;SIMULATE 2ND SECTOR PULSE

006066  004437  023740                           JSR     R4,DRSYNC               ;SIMULATE PREAMBLE
006072  104010                                   ERROR   10                      ;MR1 ERROR
006074  104011                                   ERROR   11                      ;CS1 ERROR

006076  004437  024276                           JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
006102  104012                                   ERROR   12                      ;MR1 ERROR
006104  104013                                   ERROR   13                      ;CS1 ERROR
006106  104014                                   ERROR   14                      ;RKDCYL OR RKDA IN ERROR

                                 ;;*************************************************************
                                 ;*TEST 6        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.
                                 ;*
                                 ;;*************************************************************
006110  000004                   TST6:   SCOPE
006112  012737  000012  001234           MOV     #10.,$TIMES             ;;DO 10. ITERATIONS

006120  005037  002352                   CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
006124  012762  100000  000000           MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

006132  004437  032230                   JSR     R4,LOADRK               ;LOAD "L" REGISTERS
006136  000632                           632                             ;CYLINDER 632
```

```
006140     025                          .BYTE   25              ;SECTOR 25
006141     002                          .BYTE   2               ;TRACK 2
006142  047636                          OBUFF                   ;BUFFER ADDRESS OBUFF
006144  177400                          -400                    ;WORD COUNT -400
006146  000023                          WRDATA                  ;COMMAND WRDATA

006150  004437  032014                  JSR     R4,BLDDAT       ;GO BUILD DATA
006154  000001                          1                       ;PATTERN 1

006156  004437  032274                  JSR     R4,OPSTRT       ;START THE OPERATION

006162  004437  023334                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
006166  104004                          ERROR   4               ;CS1 MISCOMPARE
006170  104005                          ERROR   5               ;MR1       "
006172  104006                          ERROR   6               ;MR2       "
006174  104007                          ERROR   7               ;MR3       "

006176  004437  023316                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

006202  004437  023740                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
006206  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
006210  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR

006212  004437  024276                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
006216  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
006220  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
006222  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

006224  012737  043403  001510          MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
006232  004437  025204                  JSR     R4,DWGPSN       ;SIMULATE GAP AND SYNC FOR WRITE
006236  104015                          ERROR   15              ;MR1 IN ERROR IN READ GAP
006240  104016                          ERROR   16              ;CS1 IN ERROR AFTER READ GAP
006242  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING SYNC
006244  104020                          ERROR   20              ;CS1 IN ERROR AFTER WRITING SYNC

006246  012737  043462  001510          MOV     #EM5021,EMW     ;SET MESSAGE FOR WRITE FAILURE
006254  004437  025604                  JSR     R4,DWRITE       ;SIMULATE WRITE DATA AND ECC
006260  104017                          ERROR   17              ;MR1 IN ERROR IN WRITING DATA OR ECC
006262  104021                          ERROR   21              ;ECC IN ERROR WHILE WRITING DATA
006264  104022                          ERROR   22              ;CS1 IN ERROR AFTER WRITING DATA AND ECC


006266  012700  000016                  MOV     #3*4+2,R0       ;SET COUNT TO END OPERATION
006272  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)  ;SET CLOCK
006300  012762  000040  000026          MOV     #DMD,RKMR1(R2)  ;CLEAR CLOCK
006306  005300                          DEC     R0              ;DEC COUNT
006310  001370                          BNE     1$              ;LOOP UNTIL 0

006312  004437  032356                  JSR     R4,GETREG       ;GET RK611 REGISTERS
006316  032737  100000  002160          BIT     #CERR,T.CS1     ;TEST IF ANY ERROR SET
006324  001415                          BEQ     3$              ;NO - SKIP TO EXIT

006326  032737  001000  002174          BIT     #COE,T.ER       ;TEST IF IT IS COE
006334  001405                          BEQ     2$              ;NO - SKIP
006336  012737  044253  001622          MOV     #E5026A,EH030   ;SET MESSAGE HEADER
006344  104030                          ERROR   30              ;"UNEXPECTED COE ERROR"
```

```
006346  000404                                  BR      3$              ;EXIT
006350  012737  044253  001632          2$:     MOV     #E5026A,EH031   ;SET HEADER
006356  104031                                  ERROR   31              ;"UNEXPECTED ERROR TESTING COE"
006360                                  3$:
                                        ;;*****************************************************************
                                        ;*TEST 7          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.
                                        ;*
                                        ;;*****************************************************************
006360  000004                          TST7:   SCOPE
006362  012737  000012  001234                  MOV     #10.,$TIMES     ;;DO 10. ITERATIONS
006370  005037  002352                          CLR     BSEDES          ;CLEAR BAD SEC DESIRED
006374  012762  100000  000000                  MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
006402  004437  032230                          JSR     R4,LOADRK       ;LOAD "L" REGISTERS
006406  000632                                  632                     ;CYLINDER 632
006410     025                                  .BYTE   25              ;SECTOR 25
006411     002                                  .BYTE   2               ;TRACK 2
006412  047636                                  OBUFF                   ;BUFFER ADDRESS OBUFF
006414  177377                                  -401                    ;WORD COUNT -401
006416  000023                                  WRDATA                  ;COMMAND WRDATA
006420  004437  032014                          JSR     R4,BLDDAT       ;GO BUILD DATA
006424  000002                                  2                       ;PATTERN 2
006426  004437  032274                          JSR     R4,OPSTRT       ;START THE OPERATION
006432  004437  023334                          JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
006436  104004                                  ERROR   4               ;CS1 MISCOMPARE
006440  104005                                  ERROR   5               ;MR1       "
006442  104006                                  ERROR   6               ;MR2       "
006444  104007                                  ERROR   7               ;MR3       "
006446  004437  023316                          JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE
006452  004437  023740                          JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
006456  104010                                  ERROR   10              ;MR1 CONTENTS IN ERROR
006460  104011                                  ERROR   11              ;CS1 CONTENTS IN ERROR
006462  004437  024276                          JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
006466  104012                                  ERROR   12              ;MR1 CONTENTS IN ERROR
006470  104013                                  ERROR   13              ;CS1 IN ERROR AFTER SEARCH
006472  104014                                  ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
006474  012737  043403  001510                  MOV     #EM5020,EMW     ;SET MESSAGE FOR WRITE FAILURE
```

```
006502  004437  025204                  JSR     R4,DWGPSN          ;SIMULATE GAP AND SYNC FOR WRITE
006506  104015                          ERROR   15                 ;MR1 IN ERROR IN READ GAP
006510  104016                          ERROR   16                 ;CS1 IN ERROR AFTER READ GAP
006512  104017                          ERROR   17                 ;MR1 IN ERROR IN WRITING SYNC
006514  104020                          ERROR   20                 ;CS1 IN ERROR AFTER WRITING SYNC

006516  012737  043462  001510          MOV     #EM5021,EMW        ;SET MESSAGE FOR WRITE FAILURE
006524  004437  025604                  JSR     R4,DWRITE          ;SIMULATE WRITE DATA AND ECC
006530  104017                          ERROR   17                 ;MR1 IN ERROR IN WRITING DATA OR ECC
006532  104021                          ERROR   21                 ;ECC IN ERROR WHILE WRITING DATA
006534  104022                          ERROR   22                 ;CS1 IN ERROR AFTER WRITING DATA AND ECC


006536  012700  000016                  MOV     #3*4+2,R0          ;SET COUNT TO COMPLETE OPERATION
006542  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)   ;RUN THE CLOCK
006550  012762  000040  000026          MOV     #DMD,RKMR1(R2)
006556  005300                          DEC     R0                 ;DEC COUNT
006560  001370                          BNE     1$                 ;LOOP UNTIL 0

006562  004437  032356                  JSR     R4,GETREG          ;GET RK611 REGS
006566  032737  001000  002174          BIT     #COE,T.ER          ;TEST IF COE SET
006574  001012                          BNE     3$                 ;YES - EXIT TEST

006576  032737  100000  002160          BIT     #CERR,T.CS1        ;TEST IF ANY OTHER ERROR SET
006604  001402                          BEQ     2$                 ;NO - SKIP
006606  104027                          ERROR   27                 ;"NO COE WHEN EXPECTED"
006610  000404                          BR      3$                 ;GO TO EXIT

006612  012737  044253  001602  2$:     MOV     #E5026A,EH026      ;SET HEADER
006620  104026                          ERROR   26                 ;"UNEXPECTED ERROR WHEN FORCING COE"

006622                          3$:

                                .SBTTL  **NPR WRITING OF MEMORY

                                ;;*****************************************************************
                                ;*TEST 10        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.
                                ;*
                                ;;*****************************************************************
006622  000004                  TST10:  SCOPE
006624  012737  000012  001234          MOV     #10.,$TIMES        ;;DO 10. ITERATIONS

006632  005037  002352                  CLR     BSEDES             ;CLEAR BAD SEC DESIRED
006636  012762  100000  000000          MOV     #CCLR,RKCS1(R2)    ;CLEAR CONTROLLER

006644  005037  002344                  CLR     ECCSRC             ;CLEAR ECC SOURCE
006650  005037  002354                  CLR     HIBITS             ;CLEAR HI ORDER BITS
006654  004437  032230                  JSR     R4,LOADRK          ;LOAD "L" REGISTERS
```

```
006660  000000                            0                         ;CYLINDER 0
006662     000                            .BYTE   0                 ;SECTOR 0
006663     000                            .BYTE   0                 ;TRACK 0
006664  046632                            IBUFF                     ;BUFFER ADDRESS IBUFF
006666  177400                            -400                      ;WORD COUNT -400
006670  000021                            RDDATA                    ;COMMAND RDDATA

006672  004437  031770                    JSR     R4,CLRIBF         ;GO CLEAR INPUT BUFFER

006676  004437  032014                    JSR     R4,BLDDAT         ;GO BUILD DATA
006702  000003                            3                         ;PATTERN 3


006704  004437  032274                    JSR     R4,OPSTRT         ;START THE OPERATION
006710  004437  023334                    JSR     R4,DISEEK         ;SIMULATE IMPLIED SEEK
006714  104004                            ERROR   4                 ;CS1 MISCOMPARE
006716  104005                            ERROR   5                 ;MR1       "
006720  104006                            ERROR   6                 ;MR2       "
006722  104007                            ERROR   7                 ;MR3       "

006724  004437  023316                    JSR     R4,DSECTR         ;SIMULATE SECTOR PULSE

006730  004437  023740                    JSR     R4,DRSYNC         ;SIMULATE HEADER PREAMBLE
006734  104010                            ERROR   10                ;MR1 CONTENTS IN ERROR
006736  104011                            ERROR   11                ;CS1 CONTENTS IN ERROR

006740  004437  024276                    JSR     R4,DHDCMP         ;SIMULATE HEADER SEARCH
006744  104012                            ERROR   12                ;MR1 CONTENTS IN ERROR
006746  104013                            ERROR   13                ;CS1 IN ERROR AFTER SEARCH
006750  104014                            ERROR   14                ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

006752  004437  026500                    JSR     R4,DRGPSN         ;GO READ GAP AND SYNC
006756  104015                            ERROR   15                ;MR1 IN ERROR READING GAP
006760  104016                            ERROR   16                ;CS1 IN ERROR AFTER READING GAP
006762  104032                            ERROR   32                ;MR1 IN ERROR READING SYNC
006764  104033                            ERROR   33                ;CS1 IN ERROR AFTER READING SYNC

006766  012737  000000  002346            MOV     #0,ECPATX         ;LOAD EXPECTED PATTERN
006774  012737  004066  002350            MOV     #4066,ECPOSX      ;LOAD EXPECTED POSITION
007002  004437  027176                    JSR     R4,DREAD          ;GO SIMULATE DATA READ
007006  000060                            60                        ;NUMBER OF WORDS TO SIMULATE
007010  104034                            ERROR   34                ;MR1 IN ERROR READING DATA OR ECC
007012  104035                            ERROR   35                ;ECC ERROR READING DATA
007014  104036                            ERROR   36                ;CS1 ERROR AFTER READING DATA OR ECC
007016  104041                            ERROR   41                ;ECC REG INCORRECT AFTER ECC READ
007020  104042                            ERROR   42                ;ERR IN ECC PAT CALC.
007022  104043                            ERROR   43                ;ERR IN ECC POS COUNT


007024  012700  047636                    MOV     #OBUFF,R0         ;SET POINTER TO GOOD DATA
007030  012701  046632                    MOV     #IBUFF,R1         ;SET POINTER TO INPUT DATA

007034  012704  000012                    MOV     #10.,R4           ;SET ERROR LIMIT COUNT
007040  005037  001236                    CLR     $ESCAPE           ;CLEAR ESCAPE
007044  005037  001220                    CLR     $TMP7             ;CLEAR ERROR REPORT SWITCH
007050  012703  000040                    MOV     #40,R3            ;SET COMPARE LIMIT
```

```
007054  005005                          CLR     R5              ;CLEAR WORD COUNTER

007056  021011                  1$:     CMP     (R0),(R1)       ;COMPARE DATA
007060  001005                          BNE     2$              ;SKIP IF NOT EQUAL
007062  005303                  4$:     DEC     R3              ;ELSE DEC WORD COUNT LIMIT
007064  001424                          BEQ     5$              ;IF 0 - SKIP TO EXIT
007066  005205                          INC     R5              ;BUMP WORD COUNT
007070  022021                          CMP     (R0)+,(R1)+     ;BUMP DATA POINTERS
007072  000771                          BR      1$              ;LOOP

007074  005304                  2$:     DEC     R4              ;DEC ERROR LIMIT
007076  001417                          BEQ     5$              ;IF 0 - SKIP

007100  011037  001162                  MOV     (R0),$REG0      ;GET GOOD WORD FOR REPORT
007104  011137  001164                  MOV     (R1),$REG1      ;GET BAD WORD
007110  010537  001166                  MOV     R5,$REG2        ;GET WORD NUMBER

007114  005737  001220                  TST     $TMP7           ;DECIDE WHICH ERROR REPORT TO USE
007120  001004                          BNE     3$              ;SKIP IF NOT 0
007122  005237  001220                  INC     $TMP7           ;ELSE BUMP $TMP7
007126  104037                          ERROR   37              ;REPORT FIRST ERROR LINE
007130  000754                          BR      4$              ;SKIP

007132  104040                  3$:     ERROR 40                ;REPORT ALL OTHER LINES
007134  000752                          BR      4$              ;SKIP

007136                          5$:

                                .SBTTL  **ECC ERROR DETECTION/CORRECTION TESTS

                                ;;*************************************************************
                                ;*TEST 11        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.
                                ;*
                                ;;*************************************************************
007136  000004                  TST11:  SCOPE
007140  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION

007146  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
007152  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER
007160  005037  002344                  CLR     ECCSRC          ;CLEAR ECC SOURCE FLAG
007164  005037  002344                  CLR     ECCSRC          ;CLEAR ECC SOURCE
007170  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
007174  004437  032230                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
007200  000000                          0                       ;CYLINDER 0
007202     000                          .BYTE   0               ;SECTOR 0
007203     000                          .BYTE   0               ;TRACK 0
007204  046632                          IBUFF                   ;BUFFER ADDRESS IBUFF
007206  177400                          -400                    ;WORD COUNT -400
```

```
007210  000021                          RDDATA                          ;COMMAND RDDATA
007212  004437  031770                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER
007216  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
007222  000004                          4                               ;PATTERN 4
007224  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION
007230  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
007234  104004                          ERROR   4                       ;CS1 MISCOMPARE
007236  104005                          ERROR   5                       ;MR1     "
007240  104006                          ERROR   6                       ;MR2     "
007242  104007                          ERROR   7                       ;MR3     "
007244  004437  023316                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE
007250  004437  023740                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
007254  104010                          ERROR   10                      ;MR1 CONTENTS IN ERROR
007256  104011                          ERROR   11                      ;CS1 CONTENTS IN ERROR
007260  004437  024276                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
007264  104012                          ERROR   12                      ;MR1 CONTENTS IN ERROR
007266  104013                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
007270  104014                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
007272  004437  026500                  JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
007276  104015                          ERROR   15                      ;MR1 IN ERROR READING GAP
007300  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
007302  104032                          ERROR   32                      ;MR1 IN ERROR READING SYNC
007304  104033                          ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC
007306  012737  000000  002346          MOV     #0,ECPATX               ;LOAD EXPECTED PATTERN
007314  012737  004066  002350          MOV     #4066,ECPOSX            ;LOAD EXPECTED POSITION
007322  004437  027176                  JSR     R4,DREAD                ;GO SIMULATE DATA READ
007326  000400                          400                             ;NUMBER OF WORDS TO SIMULATE
007330  104034                          ERROR   34                      ;MR1 IN ERROR READING DATA OR ECC
007332  104035                          ERROR   35                      ;ECC ERROR READING DATA
007334  104036                          ERROR   36                      ;CS1 ERROR AFTER READING DATA OR ECC
007336  104041                          ERROR   41                      ;ECC REG INCORRECT AFTER ECC READ
007340  104042                          ERROR   42                      ;ERR IN ECC PAT CALC.
007342  104043                          ERROR   43                      ;ERR IN ECC POS COUNT
007344  012700  002710                  MOV     #40.*37.,R0             ;SET COUNT TO EMPTY SILO
007350  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
007356  012762  000040  000026          MOV     #DMD,RKMR1(R2)
007364  005300                          DEC     R0                      ;DEC COUNT
007366  001370                          BNE     1$                      ;LOOP UNTIL 0
007370  016237  000000  002160          MOV     RKCS1(R2),T.CS1         ;GET CS1
007376  004437  032356                  JSR     R4,GETREG               ;GET 611 REGS
007402  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO BIT
007410  052737  000200  002220          BIS     #RDY,E.CS1              ;SET READY BIT
007416  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
```

```
007424  001401                          BEQ     2$                      ;YES - SKIP
007426  104056                          ERROR   56                      ;"CS1 IN ERROR AFTER READ DATA"
007430                          2$:
                                ;;*************************************************************
                                ;*TEST 12        READ DATA WITH ONE BIT BURST ERROR (PART 1)
                                ;*
                                ;*       CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                ;*       PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A READ DATA
                                ;*       OF 400 WORDS TO AN RK06 IN 26 SECTOR FORMAT,
                                ;*       CYLINDER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                ;*       AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                ;*       A GOOD HEADER, 400 DATA WORDS, AND AN ECC INDICATING
                                ;*       A ONE BIT BURST ERROR OF A 0 THAT SHOULD BE A 1.
                                ;*       VERIFY THE COUNTING OF THE POSITION
                                ;*       REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                                ;*       MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
                                ;*
                                ;;*************************************************************
007430  000004                  TST12:  SCOPE
007432  012737  000001  001234          MOV     #1,$TIMES               ;;DO 1 ITERATION

007440  005037  002352                  CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
007444  012762  100000  000000          MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

007452  005037  002354                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
007456  004437  032230                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
007462  000000                          0                               ;CYLINDER 0
007464     000                          .BYTE   0                       ;SECTOR 0
007465     000                          .BYTE   0                       ;TRACK 0
007466  046632                          IBUFF                           ;BUFFER ADDRESS IBUFF
007470  177400                          -400                            ;WORD COUNT -400
007472  000021                          RDDATA                          ;COMMAND RDDATA

007474  004437  031770                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

007500  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
007504  000007                          7                               ;PATTERN 7

007506  012700  000530                  MOV     #530,R0                 ;SET INDEX INTO BUFFER

007512  012737  177446  050636          MOV     #177446,OBUFF+1000      ;STORE ECC WORDS
007520  012737  015457  050640          MOV     #15457,OBUFF+1002
007526  012737  000001  002344          MOV     #1,ECCSRC               ;SET SOURCE TO INDICATE ECC IN BUFFER
007534  032760  000100  047636          BIT     #BIT6,OBUFF(R0)         ;TEST IF BIT SET
007542  001004                          BNE     100$                    ;YES - SKIP
007544  052760  000100  047636          BIS     #BIT6,OBUFF(R0)         ;ELSE SET BIT
007552  000403                          BR      101$
007554  042760  000100  047636  100$:   BIC     #BIT6,OBUFF(R0)         ;CLEAR BIT
007562                          101$:

007562  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION
007566  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
007572  104004                          ERROR   4                       ;CS1 MISCOMPARE
007574  104005                          ERROR   5                       ;MR1     "
007576  104006                          ERROR   6                       ;MR2     "
007600  104007                          ERROR   7                       ;MR3     "
```

```

007602  004437  023316                    JSR     R4,DSECTR            ;SIMULATE SECTOR PULSE

007606  004437  023740                    JSR     R4,DRSYNC            ;SIMULATE HEADER PREAMBLE
007612  104010                            ERROR   10                   ;MR1 CONTENTS IN ERROR
007614  104011                            ERROR   11                   ;CS1 CONTENTS IN ERROR

007616  004437  024276                    JSR     R4,DHDCMP            ;SIMULATE HEADER SEARCH
007622  104012                            ERROR   12                   ;MR1 CONTENTS IN ERROR
007624  104013                            ERROR   13                   ;CS1 IN ERROR AFTER SEARCH
007626  104014                            ERROR   14                   ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

007630  004437  026500                    JSR     R4,DRGPSN            ;GO READ GAP AND SYNC
007634  104015                            ERROR   15                   ;MR1 IN ERROR READING GAP
007636  104016                            ERROR   16                   ;CS1 IN ERROR AFTER READING GAP
007640  104032                            ERROR   32                   ;MR1 IN ERROR READING SYNC
007642  104033                            ERROR   33                   ;CS1 IN ERROR AFTER READING SYNC

007644  012737  000001  002346            MOV     #1,ECPATX            ;LOAD EXPECTED PATTERN
007652  012737  004066  002350            MOV     #4066,ECPOSX         ;LOAD EXPECTED POSITION
007660  004437  027176                    JSR     R4,DREAD             ;GO SIMULATE DATA READ
007664  000400                            400                          ;NUMBER OF WORDS TO SIMULATE
007666  104034                            ERROR   34                   ;MR1 IN ERROR READING DATA OR ECC
007670  104035                            ERROR   35                   ;ECC ERROR READING DATA
007672  104036                            ERROR   36                   ;CS1 ERROR AFTER READING DATA OR ECC
007674  104041                            ERROR   41                   ;ECC REG INCORRECT AFTER ECC READ
007676  104042                            ERROR   42                   ;ERR IN ECC PAT CALC.
007700  104043                            ERROR   43                   ;ERR IN ECC POS COUNT


007702  012737  000020  002326            MOV     #20,BITCNT           ;SET BIT COUNT FOR CORRECTION COUNT
007710  012700  000005                    MOV     #5,R0                ;SET COUNT FOR ECCPOS PASS THRU 0
007714  012701  013672                    MOV     #13672,R1            ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
007720  005037  001214                    CLR     $TMPS                ;CLEAR SWITCH

007724  004737  031616            3$:     JSR     PC,RDBIT             ;GO READ BIT
007730  004737  030462                    JSR     PC,ECCGEN            ;GENERATE ECC
007734  016237  000032  002214            MOV     RKECPT(R2),T.ECPT         ;GET PATTERN
007742  016237  000030  002212            MOV     RKECPS(R2),T.ECPS         ;GET POSITION
007750  005237  002350                    INC     ECPOSX               ;BUMP POSITION EXPECTED
007754  013737  002350  002252            MOV     ECPOSX,E.ECPS        ;SET FOR COMPARE AND REPORT
007762  023737  002254  002214            CMP     E.ECPT,T.ECPT        ;CHECK IF PATTERN CORRECT
007770  001401                            BEQ     1$                   ;YES - SKIP
007772  104042                            ERROR   42
007774  023737  002252  002212    1$:     CMP     E.ECPS,T.ECPS        ;CHECK IF POSITION CORRECT
010002  001401                            BEQ     2$                   ;YES - SKIP
010004  104043                            ERROR   43
010006  005237  002326            2$:     INC     BITCNT               ;BUMP BIT COUNT
010012  005301                            DEC     R1                   ;DEC COUNT TO ZERO IN ECCPOS
010014  001343                            BNE     3$                   ;NOT YET ZERO - LOOP
010016  005300                            DEC     R0                   ;DEC PASS THRU 0 COUNT
010020  001406                            BEQ     4$                   ;IF 0 - SKIP
010022  012737  177777  002350            MOV     #-1,ECPOSX           ;PRESET EXPECTED POS TO GO TO ZERO
010030  012701  020000                    MOV     #20000,R1            ;SET COUNT TO NEXT 0 IN ECC POS
010034  000733                            BR      3$                   ;GO DO LOOP AGAIN

```

```
010036  005737  001214                  4$:     TST     $TMP5                   ;TEST SWITCH
010042  001012                                  BNE     5$                      ;ALREADY SET - SKIP
010044  012701  005276                          MOV     #5276,R1                ;SET COUNT TO FINAL POSITION
010050  012700  000001                          MOV     #1,R0                   ;SET R0 TO RETURN AFTER ONE PASS
010054  012737  177777  002350                  MOV     #-1,ECPOSX              ;SET EXP POS TO GO TO 0 ON NEXT BIT
010062  005237  001214                          INC     $TMP5                   ;BUMP SWITCH
010066  000716                                  BR      3$                      ;GO DO LOOP AGAIN

010070  004737  031616                  5$:     JSR     PC,RDBIT                ;ONE MORE BIT TO SET READY
010074  004437  032356                          JSR     R4,GETREG               ;GET '611 REGISTERS
010100  013737  002260  002220                  MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
010106  052737  100200  002220                  BIS     #RDY!CERR,E.CS1         ;SET READY AND ERROR
010114  042737  000001  002220                  BIC     #GO,E.CS1               ;CLEAR GO
010122  023737  002160  002220                  CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
010130  001402                                  BEQ     6$                      ;YES - SKIP
010132  104044                                  ERROR   44
010134  000410                                  BR      7$
010136  032737  000100  002174          6$:     BIT     #ECH,T.ER               ;TEST IF ECC HARD ERROR SET
010144  001404                                  BEQ     7$                      ;NO - SKIP
010146  012737  100000  002234                  MOV     #DCK,E.ER               ;SET EXPECTED ERROR REG
010154  104045                                  ERROR   45
010156                                  7$:

                                        ;;*************************************************************
                                        ;*TEST 13        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 COUNTING OF THE POSITION
                                        ;*       REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                                        ;*       MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
                                        ;*
                                        ;;*************************************************************
010156  000004                          TST13:  SCOPE
010160  012737  000001  001234                  MOV     #1,$TIMES               ;;DO 1 ITERATION
010166  005037  002352                          CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
010172  012762  100000  000000                  MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

010200  005037  002354                          CLR     HIBITS                  ;CLEAR HI ORDER BITS
010204  004437  032230                          JSR     R4,LOADRK               ;LOAD "L" REGISTERS
010210  000000                                  0                               ;CYLINDER 0
010212     000                                  .BYTE   0                       ;SECTOR 0
010213     000                                  .BYTE   0                       ;TRACK 0
010214  046632                                  IBUFF                           ;BUFFER ADDRESS IBUFF
010216  177400                                  -400                            ;WORD COUNT -400
010220  000021                                  RDDATA                          ;COMMAND RDDATA

010222  004437  031770                          JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

010226  004437  032014                          JSR     R4,BLDDAT               ;GO BUILD DATA
010232  000007                                  7                               ;PATTERN 7
```

```
010234  012700  000530                    MOV     #530,R0             ;SET INDEX INTO BUFFER

010240  012737  177446  050636            MOV     #177446,OBUFF+1000       ;STORE ECC WORDS
010246  012737  015457  050640            MOV     #15457,OBUFF+1002
010254  012737  000001  002344            MOV     #1,ECCSRC           ;SET SOURCE TO INDICATE ECC IN BUFFER
010262  032760  000100  047636            BIT     #BIT6,OBUFF(R0)     ;TEST IF BIT SET
010270  001004                            BNE     100$                ;YES - SKIP
010272  052760  000100  047636            BIS     #BIT6,OBUFF(R0)     ;ELSE SET BIT
010300  000403                            BR      101$
010302  042760  000100  047636    100$:   BIC     #BIT6,OBUFF(R0)     ;CLEAR BIT
010310                            101$:

010310  004437  032274                    JSR     R4,OPSTRT           ;START THE OPERATION
010314  004437  023334                    JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
010320  104004                            ERROR   4                   ;CS1 MISCOMPARE
010322  104005                            ERROR   5                   ;MR1      "
010324  104006                            ERROR   6                   ;MR2      "
010326  104007                            ERROR   7                   ;MR3      "

010330  004437  023316                    JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

010334  004437  023740                    JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
010340  104010                            ERROR   10                  ;MR1 CONTENTS IN ERROR
010342  104011                            ERROR   11                  ;CS1 CONTENTS IN ERROR

010344  004437  024276                    JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
010350  104012                            ERROR   12                  ;MR1 CONTENTS IN ERROR
010352  104013                            ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
010354  104014                            ERROR   14                  ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

010356  004437  026500                    JSR     R4,DRGPSN           ;GO READ GAP AND SYNC
010362  104015                            ERROR   15                  ;MR1 IN ERROR READING GAP
010364  104016                            ERROR   16                  ;CS1 IN ERROR AFTER READING GAP
010366  104032                            ERROR   32                  ;MR1 IN ERROR READING SYNC
010370  104033                            ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

010372  012737  000001  002346            MOV     #1,ECPATX           ;LOAD EXPECTED PATTERN
010400  012737  004066  002350            MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
010406  004437  027176                    JSR     R4,DREAD            ;GO SIMULATE DATA READ
010412  000400                            400                         ;NUMBER OF WORDS TO SIMULATE
010414  104034                            ERROR   34                  ;MR1 IN ERROR READING DATA OR ECC
010416  104035                            ERROR   35                  ;ECC ERROR READING DATA
010420  104036                            ERROR   36                  ;CS1 ERROR AFTER READING DATA OR ECC
010422  104041                            ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
010424  104042                            ERROR   42                  ;ERR IN ECC PAT CALC.
010426  104043                            ERROR   43                  ;ERR IN ECC POS COUNT


010430  012737  000020  002326            MOV     #20,BITCNT          ;SET BIT COUNT FOR CORRECTION COUNT
010436  012700  000005                    MOV     #5,R0               ;SET COUNT FOR ECCPOS PASS THRU 0
010442  012701  013672                    MOV     #13672,R1           ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
010446  005037  001214                    CLR     $TMP5               ;CLEAR SWITCH

010452  004737  031616            3$:     JSR     PC,RDBIT            ;GO READ BIT
010456  004737  030462                    JSR     PC,ECCGEN           ;GENERATE ECC
010462  016237  000032  002214            MOV     RKECPT(R2),T.ECPT        ;GET PATTERN
```

```
010470  016237  000030  002212          MOV     RKECPS(R2),T.ECPS       ;GET POSITION
010476  005237  002350                  INC     ECPOSX                  ;BUMP POSITION EXPECTED
010502  013737  002350  002252          MOV     ECPOSX,E.ECPS           ;SET FOR COMPARE AND REPORT
010510  023737  002254  002214          CMP     E.ECPT,T.ECPT           ;CHECK IF PATTERN CORRECT
010516  001401                          BEQ     1$                      ;YES - SKIP
010520  104042                          ERROR   42
010522  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS           ;CHECK IF POSITION CORRECT
010530  001401                          BEQ     2$                      ;YES - SKIP
010532  104043                          ERROR   43
010534  005237  002326          2$:     INC     BITCNT                  ;BUMP BIT COUNT
010540  005301                          DEC     R1                      ;DEC COUNT TO ZERO IN ECCPOS
010542  001343                          BNE     3$                      ;NOT YET ZERO - LOOP
010544  005300                          DEC     R0                      ;DEC PASS THRU 0 COUNT
010546  001406                          BEQ     4$                      ;IF 0 - SKIP
010550  012737  177777  002350          MOV     #-1,ECPOSX              ;PRESET EXPECTED POS TO GO TO ZERO
010556  012701  020000                  MOV     #20000,R1               ;SET COUNT TO NEXT 0 IN ECC POS
010562  000733                          BR      3$                      ;GO DO LOOP AGAIN

010564  005737  001214          4$:     TST     $TMP5                   ;TEST SWITCH
010570  001012                          BNE     5$                      ;ALREADY SET - SKIP
010572  012701  005276                  MOV     #5276,R1                ;SET COUNT TO FINAL POSITION
010576  012700  000001                  MOV     #1,R0                   ;SET R0 TO RETURN AFTER ONE PASS
010602  012737  177777  002350          MOV     #-1,ECPOSX              ;SET EXP POS TO GO TO 0 ON NEXT BIT
010610  005237  001214                  INC     $TMP5                   ;BUMP SWITCH
010614  000716                          BR      3$                      ;GO DO LOOP AGAIN

010616  004737  031616          5$:     JSR     PC,RDBIT                ;ONE MORE BIT TO SET READY
010622  004437  032356                  JSR     R4,GETREG               ;GET '611 REGISTERS
010626  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
010634  052737  100200  002220          BIS     #RDY!CERR,E.CS1         ;SET READY AND ERROR
010642  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO
010650  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
010656  001402                          BEQ     6$                      ;YES - SKIP
010660  104044                          ERROR   44
010662  000410                          BR      7$
010664  032737  000100  002174  6$:     BIT     #ECH,T.ER               ;TEST IF ECC HARD ERROR SET
010672  001404                          BEQ     7$                      ;NO SKIP
010674  012737  100000  002234          MOV     #DCK,E.ER               ;SET EXPECTED ERROR REG
010702  104045                          ERROR   45

010704                          7$:

                                ;;*****************************************************************
                                ;*TEST 14       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.
                                ;*
```

```
                                                ;;*********************************************************************
010704  000004                                  TST14:  SCOPE
010706  012737  000001  001234                          MOV     #1,$TIMES               ;;DO 1 ITERATION

010714  005037  002352                                  CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
010720  012762  100000  000000                          MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

010726  005037  002354                                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
010732  004437  032230                                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
010736  000000                                          0                               ;CYLINDER 0
010740     000                                          .BYTE   0                       ;SECTOR 0
010741     000                                          .BYTE   0                       ;TRACK 0
010742  046632                                          IBUFF                           ;BUFFER ADDRESS IBUFF
010744  177400                                          -400                            ;WORD COUNT -400
010746  000021                                          RDDATA                          ;COMMAND RDDATA

010750  004437  031770                                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

010754  004437  032014                                  JSR     R4,BLDDAT               ;GO BUILD DATA
010760  000005                                          5                               ;PATTERN 5

010762  012700  000530                                  MOV     #530,R0                 ;SET INDEX INTO BUFFER

010766  012737  126073  050636                          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
010774  012737  151052  050640                          MOV     #151052,OBUFF+1002
011002  012737  000001  002344                          MOV     #1,ECCSRC               ;SET SOURCE TO INDICATE ECC IN BUFFER
011010  032760  000100  047636                          BIT     #BIT6,OBUFF(R0)         ;TEST IF BIT SET
011016  001004                                          BNE     100$                    ;YES - SKIP
011020  052760  000100  047636                          BIS     #BIT6,OBUFF(R0)         ;ELSE SET BIT
011026  000403                                          BR      101$
011030  042760  000100  047636          100$:           BIC     #BIT6,OBUFF(R0)         ;CLEAR BIT
011036                                  101$:

011036  004437  032274                                  JSR     R4,OPSTRT               ;START THE OPERATION
011042  004437  023334                                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
011046  104004                                          ERROR   4                       ;CS1 MISCOMPARE
011050  104005                                          ERROR   5                       ;MR1     "
011052  104006                                          ERROR   6                       ;MR2     "
011054  104007                                          ERROR   7                       ;MR3     "

011056  004437  023316                                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

011062  004437  023740                                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
011066  104010                                          ERROR   10                      ;MR1 CONTENTS IN ERROR
011070  104011                                          ERROR   11                      ;CS1 CONTENTS IN ERROR

011072  004437  024276                                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
011076  104012                                          ERROR   12                      ;MR1 CONTENTS IN ERROR
011100  104013                                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
011102  104014                                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

011104  004437  026500                                  JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
011110  104015                                          ERROR   15                      ;MR1 IN ERROR READING GAP
011112  104016                                          ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
011114  104032                                          ERROR   32                      ;MR1 IN ERROR READING SYNC
011116  104033                                          ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC
```

```
011120  012737  000001  002346          MOV     #1,ECPATX           ;LOAD EXPECTED PATTERN
011126  012737  004066  002350          MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
011134  004437  027176                  JSR     R4,DREAD            ;GO SIMULATE DATA READ
011140  000400                          400                         ;NUMBER OF WORDS TO SIMULATE
011142  104034                          ERROR   34                  ;MR1 IN ERROR READING DATA OR ECC
011144  104035                          ERROR   35                  ;ECC ERROR READING DATA
011146  104036                          ERROR   36                  ;CS1 ERROR AFTER READING DATA OR ECC
011150  104041                          ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
011152  104042                          ERROR   42                  ;ERR IN ECC PAT CALC.
011154  104043                          ERROR   43                  ;ERR IN ECC POS COUNT


011156  012737  000020  002326          MOV     #20,BITCNT          ;SET BIT COUNT FOR CORRECTION COUNT
011164  012700  000005                  MOV     #5,R0               ;SET COUNT FOR ECCPOS PASS THRU 0
011170  012701  013672                  MOV     #13672,R1           ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
011174  005037  001214                  CLR     $TMP5               ;CLEAR SWITCH

011200  004737  031616          3$:     JSR     PC,RDBIT            ;GO READ BIT
011204  004737  030462                  JSR     PC,ECCGEN           ;GENERATE ECC
011210  016237  000032  002214          MOV     RKECPT(R2),T.ECPT          ;GET PATTERN
011216  016237  000030  002212          MOV     RKECPS(R2),T.ECPS          ;GET POSITION
011224  005237  002350                  INC     ECPOSX              ;BUMP POSITION EXPECTED
011230  013737  002350  002252          MOV     ECPOSX,E.ECPS       ;SET FOR COMPARE AND REPORT
011236  023737  002254  002214          CMP     E.ECPT,T.ECPT       ;CHECK IF PATTERN CORRECT
011244  001401                          BEQ     1$                  ;YES - SKIP
011246  104042                          ERROR   42
011250  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS       ;CHECK IF POSITION CORRECT
011256  001401                          BEQ     2$                  ;YES - SKIP
011260  104043                          ERROR   43
011262  005237  002326          2$:     INC     BITCNT              ;BUMP BIT COUNT
011266  005301                          DEC     R1                  ;DEC COUNT TO ZERO IN ECCPOS
011270  001343                          BNE     3$                  ;NOT YET ZERO - LOOP
011272  005300                          DEC     R0                  ;DEC PASS THRU 0 COUNT
011274  001406                          BEQ     4$                  ;IF 0 - SKIP
011276  012737  177777  002350          MOV     #-1,ECPOSX          ;PRESET EXPECTED POS TO GO TO ZERO
011304  012701  020000                  MOV     #20000,R1           ;SET COUNT TO NEXT 0 IN ECC POS
011310  000733                          BR      3$                  ;GO DO LOOP AGAIN

011312  005737  001214          4$:     TST     $TMP5               ;TEST SWITCH
011316  001012                          BNE     5$                  ;ALREADY SET - SKIP
011320  012701  005276                  MOV     #5276,R1            ;SET COUNT TO FINAL POSITION
011324  012700  000001                  MOV     #1,R0               ;SET R0 TO RETURN AFTER ONE PASS
011330  012737  177777  002350          MOV     #-1,ECPOSX          ;SET EXP POS TO GO TO 0 ON NEXT BIT
011336  005237  001214                  INC     $TMP5               ;BUMP SWITCH
011342  000716                          BR      3$                  ;GO DO LOOP AGAIN

011344  004737  031616          5$:     JSR     PC,RDBIT            ;ONE MORE BIT TO SET READY
011350  004437  032356                  JSR     R4,GETREG           ;GET '611 REGISTERS
011354  013737  002260  002220          MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
011362  052737  100200  002220          BIS     #RDY!CERR,E.CS1     ;SET READY AND ERROR
011370  042737  000001  002220          BIC     #GO,E.CS1           ;CLEAR GO
011376  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
011404  001402                          BEQ     6$                  ;YES - SKIP
011406  104044                          ERROR   44
011410  000410                          BR      7$
```

```
 011412  032737  000100  002174  6$:      BIT     #ECH,T.ER         ;TEST IF ECC HARD ERROR SET
 011420  001404                           BEQ     7$                ;NO SKIP
 011422  012737  100000  002234           MOV     #DCK,E.ER         ;SET EXPECTED ERROR REG
 011430  104045                           ERROR   45

 011432                          7$:

                                 ;;*****************************************************************
                                 ;*TEST 15        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 POSITION
                                 ;*        REGISTER AND THE FINAL PATTERN AND POSITION INFORMATION.
                                 ;*        MAKE SURE DATA CHECK AND CONTROLLER ERROR SETS.
                                 ;*
                                 ;;*****************************************************************
 011432  000004                  TST15:   SCOPE
 011434  012737  000001  001234           MOV     #1,$TIMES         ;;DO 1 ITERATION

 011442  005037  002352                   CLR     BSEDES            ;CLEAR BAD SEC DESIRED

 011446  012762  100000  000000           MOV     #CCLR,RKCS1(R2)   ;CLEAR CONTROLLER

 011454  005037  002354                   CLR     HIBITS            ;CLEAR HI ORDER BITS
 011460  004437  032230                   JSR     R4,LOADRK         ;LOAD "L" REGISTERS
 011464  000000                           0                         ;CYLINDER 0
 011466     000                           .BYTE   0                 ;SECTOR 0
 011467     000                           .BYTE   0                 ;TRACK 0
 011470  046632                           IBUFF                     ;BUFFER ADDRESS IBUFF
 011472  177400                           -400                      ;WORD COUNT -400
 011474  000021                           RDDATA                    ;COMMAND RDDATA

 011476  004437  031770                   JSR     R4,CLRIBF         ;GO CLEAR INPUT BUFFER

 011502  004437  032014                   JSR     R4,BLDDAT         ;GO BUILD DATA
 011506  000005                           5                         ;PATTERN 5

 011510  012700  000530                   MOV     #530,R0           ;SET INDEX INTO BUFFER

 011514  012737  126073  050636           MOV     #126073,OBUFF+1000        ;STORE ECC WORDS
 011522  012737  151052  050640           MOV     #151052,OBUFF+1002
 011530  012737  000001  002344           MOV     #1,ECCSRC         ;SET SOURCE TO INDICATE ECC IN BUFFER
 011536  042760  005000  047636           BIC     #BIT9!BIT11,OBUFF(R0)    ;CREATE 11 BIT BURST ERROR
 011544  052760  050100  047636           BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
 011552  062700  000002                   ADD     #2,R0             ;BUMP TO NEXT WORD
 011556  052760  000001  047636           BIS     #BIT0,OBUFF(R0)

 011564  004437  032274                   JSR     R4,OPSTRT         ;START THE OPERATION
 011570  004437  023334                   JSR     R4,DISEEK         ;SIMULATE IMPLIED SEEK
 011574  104004                           ERROR   4                 ;CS1 MISCOMPARE
 011576  104005                           ERROR   5                 ;MR1      "
```

```
011600  104006                                 ERROR   6                       ;MR2       "
011602  104007                                 ERROR   7                       ;MR3       "

011604  004437  023316                         JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

011610  004437  023740                         JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
011614  104010                                 ERROR   10                      ;MR1 CONTENTS IN ERROR.
011616  104011                                 ERROR   11                      ;CS1 CONTENTS IN ERROR

011620  004437  024276                         JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
011624  104012                                 ERROR   12                      ;MR1 CONTENTS IN ERROR
011626  104013                                 ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
011630  104014                                 ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

011632  004437  026500                         JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
011636  104015                                 ERROR   15                      ;MR1 IN ERROR READING GAP
011640  104016                                 ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
011642  104032                                 ERROR   32                      ;MR1 IN ERROR READING SYNC
011644  104033                                 ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC

011646  012737  000001  002346                 MOV     #1,ECPATX               ;LOAD EXPECTED PATTERN
011654  012737  004066  002350                 MOV     #4066,ECPOSX            ;LOAD EXPECTED POSITION
011662  004437  027176                         JSR     R4,DREAD                ;GO SIMULATE DATA READ
011666  000400                                 400                             ;NUMBER OF WORDS TO SIMULATE
011670  104034                                 ERROR   34                      ;MR1 IN ERROR READING DATA OR ECC
011672  104035                                 ERROR   35                      ;ECC ERROR READING DATA
011674  104036                                 ERROR   36                      ;CS1 ERROR AFTER READING DATA OR ECC
011676  104041                                 ERROR   41                      ;ECC REG INCORRECT AFTER ECC READ
011700  104042                                 ERROR   42                      ;ERR IN ECC PAT CALC.
011702  104043                                 ERROR   43                      ;ERR IN ECC POS COUNT


011704  012737  000020  002326                 MOV     #20,BITCNT              ;SET BIT COUNT FOR CORRECTION COUNT
011712  012700  000005                         MOV     #5,R0                   ;SET COUNT FOR ECCPOS PASS THRU 0
011716  012701  013672                         MOV     #13672,R1               ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
011722  005037  001214                         CLR     $TMPS                   ;CLEAR SWITCH

011726  004737  031616          3$:            JSR     PC,RDBIT                ;GO READ BIT
011732  004737  030462                         JSR     PC,ECCGEN               ;GENERATE ECC
011736  016237  000032  002214                 MOV     RKECPT(R2),T.ECPT             ;GET PATTERN
011744  016237  000030  002212                 MOV     RKECPS(R2),T.ECPS             ;GET POSITION
011752  005237  002350                         INC     ECPOSX                  ;BUMP POSITION EXPECTED
011756  013737  002350  002252                 MOV     ECPOSX,E.ECPS           ;SET FOR COMPARE AND REPORT
011764  023737  002254  002214                 CMP     E.ECPT,T.ECPT           ;CHECK IF PATTERN CORRECT
011772  001401                                 BEQ     1$                      ;YES - SKIP
011774  104042                                 ERROR   42
011776  023737  002252  002212  1$:            CMP     E.ECPS,T.ECPS           ;CHECK IF POSITION CORRECT
012004  001401                                 BEQ     2$                      ;YES - SKIP
012006  104043                                 ERROR   43
012010  005237  002326          2$:            INC     BITCNT                  ;BUMP BIT COUNT
012014  005301                                 DEC     R1                      ;DEC COUNT TO ZERO IN ECCPOS
012016  001343                                 BNE     3$                      ;NOT YET ZERO - LOOP
012020  005300                                 DEC     R0                      ;DEC PASS THRU 0 COUNT
012022  001406                                 BEQ     4$                      ;IF 0 - SKIP
012024  012737  177777  002350                 MOV     #-1,ECPOSX              ;PRESET EXPECTED POS TO GO TO ZERO
012032  012701  020000                         MOV     #20000,R1               ;SET COUNT TO NEXT 0 IN ECC POS
```

```
012036  000733                           BR      3$              ;GO DO LOOP AGAIN

012040  005737  001214           4$:     TST     $TMP5           ;TEST SWITCH
012044  001012                           BNE     5$              ;ALREADY SET - SKIP
012046  012701  005310                   MOV     #5310,R1        ;SET COUNT TO FINAL POSITION
012052  012700  000001                   MOV     #1,R0           ;SET R0 TO RETURN AFTER ONE PASS
012056  012737  177777  002350           MOV     #-1,ECPOSX      ;SET EXP POS TO GO TO 0 ON NEXT BIT
012064  005237  001214                   INC     $TMP5           ;BUMP SWITCH
012070  000716                           BR      3$              ;GO DO LOOP AGAIN

012072  004737  031616           5$:     JSR     PC,RDBIT        ;ONE MORE BIT TO SET READY
012076  004437  032356                   JSR     R4,GETREG       ;GET '611 REGISTERS
012102  013737  002260  002220           MOV     L.CS1,E.CS1     ;GET EXPECTED CS1
012110  052737  100200  002220           BIS     #RDY!CERR,E.CS1 ;SET READY AND ERROR
012116  042737  000001  002220           BIC     #GO,E.CS1       ;CLEAR GO
012124  023737  002160  002220           CMP     T.CS1,E.CS1     ;CHECK IF CS1 CORRECT
012132  001402                           BEQ     6$              ;YES - SKIP
012134  104044                           ERROR   44
012136  000410                           BR      7$
012140  032737  000100  002174   6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
012146  001404                           BEQ     7$              ;NO SKIP
012150  012737  100000  002234           MOV     #DCK,E.ER       ;SET EXPECTED ERROR REG
012156  104045                           ERROR   45

012160                           7$:

                                 ;;*************************************************************
                                 ;*TEST 16        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.
                                 ;*
                                 ;*
                                 ;;*************************************************************
012160  000004                   TST16:  SCOPE
012162  012737  000001  001234           MOV     #1,$TIMES       ;;DO 1 ITERATION

012170  005037  002352                   CLR     BSEDES          ;CLEAR BAD SEC DESIRED
012174  012762  100000  000000           MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

012202  005037  002354                   CLR     HIBITS          ;CLEAR HI ORDER BITS
012206  004437  032230                   JSR     R4,LOADRK       ;LOAD "L" REGISTERS
012212  000000                           0                       ;CYLINDER 0
012214     000                           .BYTE   0               ;SECTOR 0
012215     000                           .BYTE   0               ;TRACK 0
012216  046632                           IBUFF                   ;BUFFER ADDRESS IBUFF
012220  177400                           -400                    ;WORD COUNT -400
012222  000021                           RDDATA                  ;COMMAND RDDATA

012224  004437  031770                   JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER
```

```

012230  004437  032014           JSR     R4,BLDDAT              ;GO BUILD DATA
012234  000005                   5                              ;PATTERN 5

012236  012700  000530           MOV     #530,R0                ;SET INDEX INTO BUFFER

012242  012737  126073  050636   MOV     #126073,OBUFF+1000         ;STORE ECC WORDS
012250  012737  151052  050640   MOV     #151052,OBUFF+1002
012256  012737  000001  002344   MOV     #1,ECCSRC              ;SET SOURCE TO INDICATE ECC IN BUFFER
012264  042760  005000  047636   BIC     #BIT9!BIT11,OBUFF(R0)    ;CREATE 11 BIT BURST ERROR
012272  052760  050100  047636   BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
012300  062700  000002           ADD     #2,R0                  ;BUMP TO NEXT WORD
012304  052760  000001  047636   BIS     #BIT0,OBUFF(R0)
012312  042760  000002  047636   BIC     #BIT1,OBUFF(R0)        ;MAKE IT A 12 BIT BURST ERROR

012320  004437  032274           JSR     R4,OPSTRT              ;START THE OPERATION
012324  004437  023334           JSR     R4,DISEEK              ;SIMULATE IMPLIED SEEK
012330  104004                   ERROR   4                      ;CS1 MISCOMPARE
012332  104005                   ERROR   5                      ;MR1      "
012334  104006                   ERROR   6                      ;MR2      "
012336  104007                   ERROR   7                      ;MR3      "

012340  004437  023316           JSR     R4,DSECTR              ;SIMULATE SECTOR PULSE

012344  004437  023740           JSR     R4,DRSYNC              ;SIMULATE HEADER PREAMBLE
012350  104010                   ERROR   10                     ;MR1 CONTENTS IN ERROR
012352  104011                   ERROR   11                     ;CS1 CONTENTS IN ERROR

012354  004437  024276           JSR     R4,DHDCMP              ;SIMULATE HEADER SEARCH
012360  104012                   ERROR   12                     ;MR1 CONTENTS IN ERROR
012362  104013                   ERROR   13                     ;CS1 IN ERROR AFTER SEARCH
012364  104014                   ERROR   14                     ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

012366  004437  026500           JSR     R4,DRGPSN              ;GO READ GAP AND SYNC
012372  104015                   ERROR   15                     ;MR1 IN ERROR READING GAP
012374  104016                   ERROR   16                     ;CS1 IN ERROR AFTER READING GAP
012376  104032                   ERROR   32                     ;MR1 IN ERROR READING SYNC
012400  104033                   ERROR   33                     ;CS1 IN ERROR AFTER READING SYNC

012402  012737  000001  002346   MOV     #1,ECPATX              ;LOAD EXPECTED PATTERN
012410  012737  004066  002350   MOV     #4066,ECPOSX           ;LOAD EXPECTED POSITION
012416  004437  027176           JSR     R4,DREAD               ;GO SIMULATE DATA READ
012422  000400                   400                            ;NUMBER OF WORDS TO SIMULATE
012424  104034                   ERROR   34                     ;MR1 IN ERROR READING DATA OR ECC
012426  104035                   ERROR   35                     ;ECC ERROR READING DATA
012430  104036                   ERROR   36                     ;CS1 ERROR AFTER READING DATA OR ECC
012432  104041                   ERROR   41                     ;ECC REG INCORRECT AFTER ECC READ
012434  104042                   ERROR   42                     ;ERR IN ECC PAT CALC.
012436  104043                   ERROR   43                     ;ERR IN ECC POS COUNT


012440  012737  000020  002326   MOV     #20,BITCNT             ;SET BIT COUNT FOR CORRECTION COUNT
012446  012700  000005           MOV     #5,R0                  ;SET COUNT FOR ECCPOS PASS THRU 0
012452  012701  013672           MOV     #13672,R1              ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
012456  005037  001214           CLR     $TMP5                  ;CLEAR SWITCH

```

```
012462  004737  031616          3$:     JSR     PC,RDBIT                ;GO READ BIT
012466  004737  030462                  JSR     PC,ECCGEN               ;GENERATE ECC
012472  016237  000032  002214          MOV     RKECPT(R2),T.ECPT               ;GET PATTERN
012500  016237  000030  002212          MOV     RKECPS(R2),T.ECPS               ;GET POSITION
012506  005237  002350                  INC     ECPOSX                  ;BUMP POSITION EXPECTED
012512  013737  002350  002252          MOV     ECPOSX,E.ECPS           ;SET FOR COMPARE AND REPORT
012520  023737  002254  002214          CMP     E.ECPT,T.ECPT           ;CHECK IF PATTERN CORRECT
012526  001401                          BEQ     1$                      ;YES - SKIP
012530  104042                          ERROR   42
012532  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS           ;CHECK IF POSITION CORRECT
012540  001401                          BEQ     2$                      ;YES - SKIP
012542  104043                          ERROR   43
012544  005237  002326          2$:     INC     BITCNT                  ;BUMP BIT COUNT
012550  005301                          DEC     R1                      ;DEC COUNT TO ZERO IN ECCPOS
012552  001343                          BNE     3$                      ;NOT YET ZERO - LOOP
012554  005300                          DEC     R0                      ;DEC PASS THRU 0 COUNT
012556  001406                          BEQ     4$                      ;IF 0 - SKIP
012560  012737  177777  002350          MOV     #-1,ECPOSX              ;PRESET EXPECTED POS TO GO TO ZERO
012566  012701  020000                  MOV     #20000,R1               ;SET COUNT TO NEXT 0 IN ECC POS
012572  000733                          BR      3$                      ;GO DO LOOP AGAIN

012574  005737  001214          4$:     TST     $TMP5                   ;TEST SWITCH
012600  001012                          BNE     5$                      ;ALREADY SET - SKIP
012602  012701  010041                  MOV     #10041,R1               ;SET COUNT TO FINAL POSITION
012606  012700  000001                  MOV     #1,R0                   ;SET R0 TO RETURN AFTER ONE PASS
012612  012737  177777  002350          MOV     #-1,ECPOSX              ;SET EXP POS TO GO TO 0 ON NEXT BIT
012620  005237  001214                  INC     $TMP5                   ;BUMP SWITCH
012624  000716                          BR      3$                      ;GO DO LOOP AGAIN

012626  004737  031616          5$:     JSR     PC,RDBIT                ;ONE MORE BIT TO SET READY
012632  004437  032356                  JSR     R4,GETREG               ;GET '611 REGISTERS
012636  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
012644  052737  100200  002220          BIS     #RDY!CERR,E.CS1         ;SET READY AND ERROR
012652  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO
012660  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
012666  001402                          BEQ     6$                      ;YES - SKIP
012670  104044                          ERROR   44
012672  000410                          BR      7$
012674  032737  000100  002174  6$:     BIT     #ECH,T.ER               ;TEST IF ECC HARD ERROR SET
012702  001004                          BNE     7$                      ;YES -  SKIP
012704  012737  100100  002234          MOV     #DCK!ECH,E.ER           ;SET EXPECTED ERROR REG
012712  104045                          ERROR   45

012714                          7$:

                                ;;*************************************************************
                                ;*TEST 17        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.
                                   ;*
                                   ;;*******************************************************************
012714  000004                     TST17:    SCOPE
012716  012737  000001  001234               MOV     #1,$TIMES            ;;DO 1 ITERATION

012724  005037  002352                       CLR     BSEDES               ;CLEAR BAD SEC DESIRED
012730  012762  100000  000000               MOV     #CCLR,RKCS1(R2)      ;CLEAR CONTROLLER

012736  005037  002354                       CLR     HIBITS               ;CLEAR HI ORDER BITS
012742  004437  032230                       JSR     R4,LOADRK            ;LOAD "L" REGISTERS
012746  000000                               0                            ;CYLINDER 0
012750     000                               .BYTE   0                    ;SECTOR 0
012751     000                               .BYTE   0                    ;TRACK 0
012752  046632                               IBUFF                        ;BUFFER ADDRESS IBUFF
012754  177400                               -400                         ;WORD COUNT -400
012756  000021                               RDDATA                       ;COMMAND RDDATA

012760  004437  031770                       JSR     R4,CLRIBF            ;GO CLEAR INPUT BUFFER

012764  004437  032014                       JSR     R4,BLDDAT            ;GO BUILD DATA
012770  000005                               5                            ;PATTERN 5

012772  012700  000530                       MOV     #530,R0              ;SET INDEX INTO BUFFER

012776  012737  126073  050636               MOV     #126073,OBUFF+1000        ;STORE ECC WORDS
013004  012737  151052  050640               MOV     #151052,OBUFF+1002
013012  012737  000001  002344               MOV     #1,ECCSRC            ;SET SOURCE TO INDICATE ECC IN BUFFER
013020  042760  005000  047636               BIC     #BIT9!BIT11,OBUFF(R0)    ;CREATE 11 BIT BURST ERROR
013026  052760  050100  047636               BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
013034  062700  000002                       ADD     #2,R0                ;BUMP TO NEXT WORD
013040  052760  000001  047636               BIS     #BIT0,OBUFF(R0)
013046  052760  010000  047636               BIS     #BIT12,OBUFF(R0) ;MAKE IT A 23 BIT BURST ERROR

013054  004437  032274                       JSR     R4,OPSTRT            ;START THE OPERATION
013060  004437  023334                       JSR     R4,DISEEK            ;SIMULATE IMPLIED SEEK
013064  104004                               ERROR   4                    ;CS1 MISCOMPARE
013066  104005                               ERROR   5                    ;MR1      "
013070  104006                               ERROR   6                    ;MR2      "
013072  104007                               ERROR   7                    ;MR3      "

013074  004437  023316                       JSR     R4,DSECTR            ;SIMULATE SECTOR PULSE

013100  004437  023740                       JSR     R4,DRSYNC            ;SIMULATE HEADER PREAMBLE
013104  104010                               ERROR   10                   ;MR1 CONTENTS IN ERROR
013106  104011                               ERROR   11                   ;CS1 CONTENTS IN ERROR

013110  004437  024276                       JSR     R4,DHDCMP            ;SIMULATE HEADER SEARCH
013114  104012                               ERROR   12                   ;MR1 CONTENTS IN ERROR
013116  104013                               ERROR   13                   ;CS1 IN ERROR AFTER SEARCH
013120  104014                               ERROR   14                   ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

013122  004437  026500                       JSR     R4,DRGPSN            ;GO READ GAP AND SYNC
013126  104015                               ERROR   15                   ;MR1 IN ERROR READING GAP
013130  104016                               ERROR   16                   ;CS1 IN ERROR AFTER READING GAP
013132  104032                               ERROR   32                   ;MR1 IN ERROR READING SYNC
```

```
013134  104033                          ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

013136  012737  000001  002346          MOV     #1,ECPATX           ;LOAD EXPECTED PATTERN
013144  012737  004066  002350          MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
013152  004437  027176                  JSR     R4,DREAD            ;GO SIMULATE DATA READ
013156  000400                          400                         ;NUMBER OF WORDS TO SIMULATE
013160  104034                          ERROR   34                  ;MR1 IN ERROR READING DATA OR ECC
013162  104035                          ERROR   35                  ;ECC ERROR READING DATA
013164  104036                          ERROR   36                  ;CS1 ERROR AFTER READING DATA OR ECC
013166  104041                          ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
013170  104042                          ERROR   42                  ;ERR IN ECC PAT CALC.
013172  104043                          ERROR   43                  ;ERR IN ECC POS COUNT


013174  012737  000020  002326          MOV     #20,BITCNT          ;SET BIT COUNT FOR CORRECTION COUNT
013202  012700  000005                  MOV     #5,R0               ;SET COUNT FOR ECCPOS PASS THRU 0
013206  012701  013672                  MOV     #13672,R1           ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
013212  005037  001214                  CLR     $TMPS               ;CLEAR SWITCH

013216  004737  031616          3$:     JSR     PC,RDBIT            ;GO READ BIT
013222  004737  030462                  JSR     PC,ECCGEN           ;GENERATE ECC
013226  016237  000032  002214          MOV     RKECPT(R2),T.ECPT         ;GET PATTERN
013234  016237  000030  002212          MOV     RKECPS(R2),T.ECPS         ;GET POSITION
013242  005237  002350                  INC     ECPOSX              ;BUMP POSITION EXPECTED
013246  013737  002350  002252          MOV     ECPOSX,E.ECPS       ;SET FOR COMPARE AND REPORT
013254  023737  002254  002214          CMP     E.ECPT,T.ECPT       ;CHECK IF PATTERN CORRECT
013262  001401                          BEQ     1$                  ;YES - SKIP
013264  104042                          ERROR   42
013266  023737  002252  002212  1$:     CMP     E.ECPS,T.ECPS       ;CHECK IF POSITION CORRECT
013274  001401                          BEQ     2$                  ;YES - SKIP
013276  104043                          ERROR   43
013300  005237  002326          2$:     INC     BITCNT              ;BUMP BIT COUNT
013304  005301                          DEC     R1                  ;DEC COUNT TO ZERO IN ECCPOS
013306  001343                          BNE     3$                  ;NOT YET ZERO - LOOP
013310  005300                          DEC     R0                  ;DEC PASS THRU 0 COUNT
013312  001406                          BEQ     4$                  ;IF 0 - SKIP
013314  012737  177777  002350          MOV     #-1,ECPOSX          ;PRESET EXPECTED POS TO GO TO ZERO
013322  012701  020000                  MOV     #20000,R1           ;SET COUNT TO NEXT 0 IN ECC POS
013326  000733                          BR      3$                  ;GO DO LOOP AGAIN

013330  005737  001214          4$:     TST     $TMPS               ;TEST SWITCH
013334  001012                          BNE     5$                  ;ALREADY SET - SKIP
013336  012701  010041                  MOV     #10041,R1           ;SET COUNT TO FINAL POSITION
013342  012700  000001                  MOV     #1,R0               ;SET R0 TO RETURN AFTER ONE PASS
013346  012737  177777  002350          MOV     #-1,ECPOSX          ;SET EXP POS TO GO TO 0 ON NEXT BIT
013354  005237  001214                  INC     $TMPS               ;BUMP SWITCH
013360  000716                          BR      3$                  ;GO DO LOOP AGAIN

013362  004737  031616          5$:     JSR     PC,RDBIT            ;ONE MORE BIT TO SET READY
013366  004437  032356                  JSR     R4,GETREG           ;GET '611 REGISTERS
013372  013737  002260  002220          MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
013400  052737  100200  002220          BIS     #RDY!CERR,E.CS1     ;SET READY AND ERROR
013406  042737  000001  002220          BIC     #GO,E.CS1           ;CLEAR GO
013414  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
013422  001402                          BEQ     6$                  ;YES - SKIP
013424  104044                          ERROR   44
```

```
        013426  000410                          BR      7$
        013430  032737  000100  002174  6$:     BIT     #ECH,T.ER       ;TEST IF ECC HARD ERROR SET
        013436  001004                          BNE     7$              ;YES - SKIP
        013440  012737  100100  002234          MOV     #DCK!ECH,E.ER   ;SET EXPECTED ERROR REG
        013446  104045                          ERROR   45

        013450                          7$:

                                        ;;*************************************************************
                                        ;*TEST 20        READ DATA WITH 1 BIT ECC WORD 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 1 BIT ERROR IN ECC WORD 2, BIT 6.  VERIFY THE
                                        ;*       COUNTING OF THE POSITION REGISTER.  MAKE SURE DATA
                                        ;*       CHECK AND CONTROLLER ERROR SETS.
                                        ;*
                                        ;;*************************************************************
        013450  000004                  TST20:  SCOPE
        013452  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION

        013460  005037  002352                  CLR     BSEDES          ;CLEAR BAD SEC DESIRED
        013464  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

        013472  005037  002354                  CLR     HIBITS          ;CLEAR HI ORDER BITS
        013476  004437  032230                  JSR     R4,LOADRK       ;LOAD "L" REGISTERS
        013502  000000                          0                       ;CYLINDER 0
        013504     000                          .BYTE   0               ;SECTOR 0
        013505     000                          .BYTE   0               ;TRACK 0
        013506  046632                          IBUFF                   ;BUFFER ADDRESS IBUFF
        013510  177400                          -400                    ;WORD COUNT -400
        013512  000021                          RDDATA                  ;COMMAND RDDATA

        013514  004437  031770                  JSR     R4,CLRIBF       ;GO CLEAR INPUT BUFFER

        013520  004437  032014                  JSR     R4,BLDDAT       ;GO BUILD DATA
        013524  000005                          5                       ;PATTERN 5


        013526  012700  001002                  MOV     #1002,R0        ;SET INDEX INTO BUFFER
        013532  012737  126073  050636          MOV     #126073,OBUFF+1000      ;STORE ECC WORDS
        013540  012737  151052  050640          MOV     #151052,OBUFF+1002
        013546  012737  000001  002344          MOV     #1,ECCSRC       ;SET SOURCE TO INDICATE ECC IN BUFFER
        013554  032760  000100  047636          BIT     #BIT6,OBUFF(R0) ;TEST IF BIT SET
        013562  001004                          BNE     100$            ;YES - SKIP
        013564  052760  000100  047636          BIS     #BIT6,OBUFF(R0) ;ELSE SET BIT
        013572  000403                          BR      101$
        013574  042760  000100  047636  100$:   BIC     #BIT6,OBUFF(R0) ;CLEAR BIT
        013602                          101$:

        013602  004437  032274                  JSR     R4,OPSTRT       ;START THE OPERATION
        013606  004437  023334                  JSR     R4,DISEEK       ;SIMULATE IMPLIED SEEK
```

```
 013612  104004                            ERROR   4                  ;CS1 MISCOMPARE
 013614  104005                            ERROR   5                  ;MR1      "
 013616  104006                            ERROR   6                  ;MR2      "
 013620  104007                            ERROR   7                  ;MR3      "

 013622  004437  023316                    JSR     R4,DSECTR          ;SIMULATE SECTOR PULSE

 013626  004437  023740                    JSR     R4,DRSYNC          ;SIMULATE HEADER PREAMBLE
 013632  104010                            ERROR   10                 ;MR1 CONTENTS IN ERROR
 013634  104011                            ERROR   11                 ;CS1 CONTENTS IN ERROR

 013636  004437  024276                    JSR     R4,DHDCMP          ;SIMULATE HEADER SEARCH
 013642  104012                            ERROR   12                 ;MR1 CONTENTS IN ERROR
 013644  104013                            ERROR   13                 ;CS1 IN ERROR AFTER SEARCH
 013646  104014                            ERROR   14                 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

 013650  004437  026500                    JSR     R4,DRGPSN          ;GO READ GAP AND SYNC
 013654  104015                            ERROR   15                 ;MR1 IN ERROR READING GAP
 013656  104016                            ERROR   16                 ;CS1 IN ERROR AFTER READING GAP
 013660  104032                            ERROR   32                 ;MR1 IN ERROR READING SYNC
 013662  104033                            ERROR   33                 ;CS1 IN ERROR AFTER READING SYNC

 013664  012737  000001  002346            MOV     #1,ECPATX          ;LOAD EXPECTED PATTERN
 013672  012737  004066  002350            MOV     #4066,ECPOSX       ;LOAD EXPECTED POSITION
 013700  004437  027176                    JSR     R4,DREAD           ;GO SIMULATE DATA READ
 013704  000400                            400                        ;NUMBER OF WORDS TO SIMULATE
 013706  104034                            ERROR   34                 ;MR1 IN ERROR READING DATA OR ECC
 013710  104035                            ERROR   35                 ;ECC ERROR READING DATA
 013712  104036                            ERROR   36                 ;CS1 ERROR AFTER READING DATA OR ECC
 013714  104041                            ERROR   41                 ;ECC REG INCORRECT AFTER ECC READ
 013716  104042                            ERROR   42                 ;ERR IN ECC PAT CALC.
 013720  104043                            ERROR   43                 ;ERR IN ECC POS COUNT


 013722  012737  000020  002326            MOV     #20,BITCNT         ;SET BIT COUNT FOR CORRECTION COUNT
 013730  012700  000005                    MOV     #5,R0              ;SET COUNT FOR ECCPOS PASS THRU 0
 013734  012701  013672                    MOV     #13672,R1          ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
 013740  005037  001214                    CLR     $TMPS              ;CLEAR SWITCH

 013744  004737  031616          3$:       JSR     PC,RDBIT           ;GO READ BIT
 013750  004737  030462                    JSR     PC,ECCGEN          ;GENERATE ECC
 013754  016237  000032  002214            MOV     RKECPT(R2),T.ECPT          ;GET PATTERN
 013762  016237  000030  002212            MOV     RKECPS(R2),T.ECPS          ;GET POSITION
 013770  005237  002350                    INC     ECPOSX             ;BUMP POSITION EXPECTED
 013774  013737  002350  002252            MOV     ECPOSX,E.ECPS      ;SET FOR COMPARE AND REPORT
 014002  023737  002254  002214            CMP     E.ECPT,T.ECPT      ;CHECK IF PATTERN CORRECT
 014010  001401                            BEQ     1$                 ;YES - SKIP
 014012  104042                            ERROR   42
 014014  023737  002252  002212  1$:       CMP     E.ECPS,T.ECPS      ;CHECK IF POSITION CORRECT
 014022  001401                            BEQ     2$                 ;YES - SKIP
 014024  104043                            ERROR   43
 014026  005237  002326          2$:       INC     BITCNT             ;BUMP BIT COUNT
 014032  005301                            DEC     R1                 ;DEC COUNT TO ZERO IN ECCPOS
 014034  001343                            BNE     3$                 ;NOT YET ZERO - LOOP
 014036  005300                            DEC     R0                 ;DEC PASS THRU 0 COUNT
 014040  001406                            BEQ     4$                 ;IF 0 - SKIP
```

```
014042  012737  177777  002350          MOV     #-1,ECPOSX          ;PRESET EXPECTED POS TO GO TO ZERO
014050  012701  020000                  MOV     #20000,R1           ;SET COUNT TO NEXT 0 IN ECC POS
014054  000733                          BR      3$                  ;GO DO LOOP AGAIN

014056  005737  001214          4$:     TST     $TMP5               ;TEST SWITCH
014062  001012                          BNE     5$                  ;ALREADY SET - SKIP
014064  012701  010016                  MOV     #10016,R1           ;SET COUNT TO FINAL POSITION
014070  012700  000001                  MOV     #1,R0               ;SET R0 TO RETURN AFTER ONE PASS
014074  012737  177777  002350          MOV     #-1,ECPOSX          ;SET EXP POS TO GO TO 0 ON NEXT BIT
014102  005237  001214                  INC     $TMP5               ;BUMP SWITCH
014106  000716                          BR      3$                  ;GO DO LOOP AGAIN

014110  004737  031616          5$:     JSR     PC,RDBIT            ;ONE MORE BIT TO SET READY
014114  004437  032356                  JSR     R4,GETREG           ;GET '611 REGISTERS
014120  013737  002260  002220          MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
014126  052737  100200  002220          BIS     #RDY!CERR,E.CS1     ;SET READY AND ERROR
014134  042737  000001  002220          BIC     #GO,E.CS1           ;CLEAR GO
014142  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
014150  001402                          BEQ     6$                  ;YES - SKIP
014152  104044                          ERROR   44
014154  000410                          BR      7$
014156  032737  000100  002174  6$:     BIT     #ECH,T.ER           ;TEST IF ECC HARD ERROR SET
014164  001404                          BEQ     7$                  ;YES - SKIP
014166  012737  100000  002234          MOV     #DCK,E.ER           ;SET EXPECTED ERROR REG
014174  104045                          ERROR   45

014176                          7$:

                                ;;*****************************************************************
                                ;*TEST 21        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.
                                ;*
                                ;*****************************************************************
014176  000004                  TST21:  SCOPE
014200  012737  000012  001234          MOV     #10.,$TIMES         ;;DO 10. ITERATIONS

014206  005037  002352                  CLR     BSEDES              ;CLEAR BAD SEC DESIRED
014212  012762  100000  000000          MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER

014220  005037  002344                  CLR     ECCSRC              ;CLEAR ECC SOURCE
014224  005037  002354                  CLR     HIBITS              ;CLEAR HI ORDER BITS
014230  004437  032230                  JSR     R4,LOADRK           ;LOAD "L" REGISTERS
014234  000000                          0                           ;CYLINDER 0
014236  000                             .BYTE   0                   ;SECTOR 0
014237  000                             .BYTE   0                   ;TRACK 0
014240  046632                          IBUFF                       ;BUFFER ADDRESS IBUFF
014242  177400                          -400                        ;WORD COUNT -400
014244  010021                          RDDATA!CFMT                 ;COMMAND RDDATA!CFMT
```

```
 014246  004437  031770                  JSR     R4,CLRIBF         ;GO CLEAR INPUT BUFFER

 014252  004437  032014                  JSR     R4,BLDDAT         ;GO BUILD DATA
 014256  000003                          3                         ;PATTERN 3


 014260  004437  032274                  JSR     R4,GPSTRT         ;START THE OPERATION
 014264  004437  023334                  JSR     R4,DISEEK         ;SIMULATE IMPLIED SEEK
 014270  104004                          ERROR   4                 ;CS1 MISCOMPARE
 014272  104005                          ERROR   5                 ;MR1      "
 014274  104006                          ERROR   6                 ;MR2      "
 014276  104007                          ERROR   7                 ;MR3      "

 014300  004437  023316                  JSR     R4,DSECTR         ;SIMULATE SECTOR PULSE

 014304  004437  023740                  JSR     R4,DRSYNC         ;SIMULATE HEADER PREAMBLE
 014310  104010                          ERROR   10                ;MR1 CONTENTS IN ERROR
 014312  104011                          ERROR   11                ;CS1 CONTENTS IN ERROR

 014314  004437  024276                  JSR     R4,DHDCMP         ;SIMULATE HEADER SEARCH
 014320  104012                          ERROR   12                ;MR1 CONTENTS IN ERROR
 014322  104013                          ERROR   13                ;CS1 IN ERROR AFTER SEARCH
 014324  104014                          ERROR   14                ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

 014326  004437  026500                  JSR     R4,DRGPSN         ;GO READ GAP AND SYNC
 014332  104015                          ERROR   15                ;MR1 IN ERROR READING GAP
 014334  104016                          ERROR   16                ;CS1 IN ERROR AFTER READING GAP
 014336  104032                          ERROR   32                ;MR1 IN ERROR READING SYNC
 014340  104033                          ERROR   33                ;CS1 IN ERROR AFTER READING SYNC

 014342  012737  000000  002346          MOV     #0,ECPATX         ;LOAD EXPECTED PATTERN
 014350  012737  005066  002350          MOV     #5066,ECPOSX      ;LOAD EXPECTED POSITION
 014356  004437  027176                  JSR     R4,DREAD          ;GO SIMULATE DATA READ
 014362  000060                          60                        ;NUMBER OF WORDS TO SIMULATE
 014364  104034                          ERROR   34                ;MR1 IN ERROR READING DATA OR ECC
 014366  104035                          ERROR   35                ;ECC ERROR READING DATA
 014370  104036                          ERROR   36                ;CS1 ERROR AFTER READING DATA OR ECC
 014372  104041                          ERROR   41                ;ECC REG INCORRECT AFTER ECC READ
 014374  104042                          ERROR   42                ;ERR IN ECC PAT CALC.
 014376  104043                          ERROR   43                ;ERR IN ECC POS COUNT


 014400  012700  047636                  MOV     #OBUFF,R0         ;SET POINTER TO GOOD DATA
 014404  012701  046632                  MOV     #IBUFF,R1         ;SET POINTER TO INPUT DATA

 014410  012704  000012                  MOV     #10.,R4           ;SET ERROR LIMIT COUNT
 014414  005037  001236                  CLR     $ESCAPE           ;CLEAR ESCAPE
 014420  005037  001220                  CLR     $TMP7             ;CLEAR ERROR REPORT SWITCH
 014424  012703  000040                  MOV     #40,R3            ;SET COMPARE LIMIT
 014430  005005                          CLR     R5                ;CLEAR WORD COUNTER

 014432  021011                  1$:     CMP     (R0),(R1)         ;COMPARE DATA
 014434  001005                          BNE     2$                ;SKIP IF NOT EQUAL
 014436  005303                  4$:     DEC     R3                ;ELSE DEC WORD COUNT LIMIT
 014440  001424                          BEQ     5$                ;IF 0 - SKIP TO EXIT
 014442  005205                          INC     R5                ;BUMP WORD COUNT
```

```
 014444  022021                                   CMP     (R0)+,(R1)+          ;BUMP DATA POINTERS
 014446  000771                                   BR      1$                   ;LOOP

 014450  005304                           2$:     DEC     R4                   ;DEC ERROR LIMIT
 014452  001417                                   BEQ     5$                   ;IF 0 - SKIP

 014454  011037  001162                           MOV     (R0),$REG0           ;GET GOOD WORD FOR REPORT
 014460  011137  001164                           MOV     (R1),$REG1           ;GET BAD WORD
 014464  010537  001166                           MOV     R5,$REG2             ;GET WORD NUMBER

 014470  005737  001220                           TST     $TMP7                ;DECIDE WHICH ERROR REPORT TO USE
 014474  001004                                   BNE     3$                   ;SKIP IF NOT 0
 014476  005237  001220                           INC     $TMP7                ;ELSE BUMP $TMP7
 014502  104037                                   ERROR   37                   ;REPORT FIRST ERROR LINE
 014504  000754                                   BR      4$                   ;SKIP

 014506  104040                           3$:     ERROR 40                     ;REPORT ALL OTHER LINES
 014510  000752                                   BR      4$                   ;SKIP

 014512                                   5$:


                                          ;;*************************************************************
                                          ;*TEST 22         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.
                                          ;*
                                          ;;*************************************************************
 014512  000004                           TST22:  SCOPE
 014514  012737  000001  001234                   MOV     #1,$TIMES            ;;DO 1 ITERATION

 014522  005037  002352                           CLR     BSEDES               ;CLEAR BAD SEC DESIRED
 014526  012762  100000  000000                   MOV     #CCLR,RKCS1(R2)      ;CLEAR CONTROLLER
 014534  005037  002344                           CLR     ECCSRC               ;CLEAR ECC SOURCE FLAG
 014540  005037  002344                           CLR     ECCSRC               ;CLEAR ECC SOURCE
 014544  005037  002354                           CLR     HIBITS               ;CLEAR HI ORDER BITS
 014550  004437  032230                           JSR     R4,LOADRK            ;LOAD "L" REGISTERS
 014554  000000                                   0                            ;CYLINDER 0
 014556     000                                   .BYTE   0                    ;SECTOR 0
 014557     000                                   .BYTE   0                    ;TRACK 0
 014560  046632                                   IBUFF                        ;BUFFER ADDRESS IBUFF
 014562  177400                                   -400                         ;WORD COUNT -400
 014564  010021                                   RDDATA!CFMT                       ;COMMAND RDDATA!CFMT

 014566  004437  031770                           JSR     R4,CLRIBF            ;GO CLEAR INPUT BUFFER

 014572  004437  032014                           JSR     R4,BLDDAT            ;GO BUILD DATA
 014576  000004                                   4                            ;PATTERN 4


 014600  004437  032274                           JSR     R4,OPSTRT            ;START THE OPERATION
```

```
014604  004437  023334                  JSR     R4,DISEEK          ;SIMULATE IMPLIED SEEK
014610  104004                          ERROR   4                  ;CS1 MISCOMPARE
014612  104005                          ERROR   5                  ;MR1      "
014614  104006                          ERROR   6                  ;MR2      "
014616  104007                          ERROR   7                  ;MR3      "
014620  004437  023316                  JSR     R4,DSECTR          ;SIMULATE SECTOR PULSE
014624  004437  023740                  JSR     R4,DRSYNC          ;SIMULATE HEADER PREAMBLE
014630  104010                          ERROR   10                 ;MR1 CONTENTS IN ERROR
014632  104011                          ERROR   11                 ;CS1 CONTENTS IN ERROR
014634  004437  024276                  JSR     R4,DHDCMP          ;SIMULATE HEADER SEARCH
014640  104012                          ERROR   12                 ;MR1 CONTENTS IN ERROR
014642  104013                          ERROR   13                 ;CS1 IN ERROR AFTER SEARCH
014644  104014                          ERROR   14                 ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
014646  004437  026500                  JSR     R4,DRGPSN          ;GO READ GAP AND SYNC
014652  104015                          ERROR   15                 ;MR1 IN ERROR READING GAP
014654  104016                          ERROR   16                 ;CS1 IN ERROR AFTER READING GAP
014656  104032                          ERROR   32                 ;MR1 IN ERROR READING SYNC
014660  104033                          ERROR   33                 ;CS1 IN ERROR AFTER READING SYNC
014662  012737  000000  002346          MOV     #0,ECPATX          ;LOAD EXPECTED PATTERN
014670  012737  005066  002350          MOV     #5066,ECPOSX       ;LOAD EXPECTED POSITION
014676  004437  027176                  JSR     R4,DREAD           ;GO SIMULATE DATA READ
014702  000400                          400                        ;NUMBER OF WORDS TO SIMULATE
014704  104034                          ERROR   34                 ;MR1 IN ERROR READING DATA OR ECC
014706  104035                          ERROR   35                 ;ECC ERROR READING DATA
014710  104036                          ERROR   36                 ;CS1 ERROR AFTER READING DATA OR ECC
014712  104041                          ERROR   41                 ;ECC REG INCORRECT AFTER ECC READ
014714  104042                          ERROR   42                 ;ERR IN ECC PAT CALC.
014716  104043                          ERROR   43                 ;ERR IN ECC POS COUNT
014720  012700  003100                  MOV     #40.*40.,R0        ;SET COUNT TO EMPTY SILO
014724  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
014732  012762  000040  000026          MOV     #DMD,RKMR1(R2)
014740  005300                          DEC     R0                 ;DEC COUNT
014742  001370                          BNE     1$                 ;LOOP UNTIL 0
014744  004437  032356                  JSR     R4,GETREG          ;GET 611 REGS
014750  013737  002260  002220          MOV     L.CS1,E.CS1        ;GET EXPECTED CS1
014756  042737  000001  002220          BIC     #GO,E.CS1          ;CLEAR GO BIT
014764  052737  000200  002220          BIS     #RDY,E.CS1         ;SET READY BIT
014772  023737  002160  002220          CMP     T.CS1,E.CS1        ;CHECK IF CS1 CORRECT
015000  001401                          BEQ     2$                 ;YES - SKIP
015002  104056                          ERROR   56                 ;"CS1 IN ERROR AFTER READ DATA"
015004                          2$:
                                ;;*************************************************************
                                ;*TEST 23        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.
                                        ;*
                                        ;;*************************************************************
015004  000004                          TST23:  SCOPE
015006  012737  000001  001234                  MOV     #1,$TIMES               ;;DO 1 ITERATION

015014  005037  002352                          CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
015020  012762  100000  000000                  MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

015026  005037  002354                          CLR     HIBITS                  ;CLEAR HI ORDER BITS
015032  004437  032230                          JSR     R4,LOADRK               ;LOAD "L" REGISTERS
015036  000000                                  0                               ;CYLINDER 0
015040     000                                  .BYTE   0                       ;SECTOR 0
015041     000                                  .BYTE   0                       ;TRACK 0
015042  046632                                  IBUFF                           ;BUFFER ADDRESS IBUFF
015044  177400                                  -400                            ;WORD COUNT -400
015046  010021                                  RDDATA!CFMT                             ;COMMAND RDDATA!CFMT

015050  004437  031770                          JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

015054  004437  032014                          JSR     R4,BLDDAT               ;GO BUILD DATA
015060  000007                                  7                               ;PATTERN 7


015062  012700  000734                          MOV     #734,R0                 ;SET INDEX INTO BUFFER
015066  012737  006030  050636                  MOV     #6030,OBUFF+1000                ;STORE ECC WORDS
015074  012737  045070  050640                  MOV     #45070,OBUFF+1002
015102  012737  000001  002344                  MOV     #1,ECCSRC               ;SET SOURCE TO INDICATE ECC IN BUFFER
015110  032760  000100  047636                  BIT     #BIT6,OBUFF(R0)         ;TEST IF BIT SET
015116  001004                                  BNE     100$                    ;YES - SKIP
015120  052760  000100  047636                  BIS     #BIT6,OBUFF(R0)         ;ELSE SET BIT
015126  000403                                  BR      101$
015130  042760  000100  047636          100$:   BIC     #BIT6,OBUFF(R0)         ;CLEAR BIT
015136                                  101$:

015136  004437  032274                          JSR     R4,OPSTRT               ;START THE OPERATION
015142  004437  023334                          JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
015146  104004                                  ERROR   4                       ;CS1 MISCOMPARE
015150  104005                                  ERROR   5                       ;MR1     "
015152  104006                                  ERROR   6                       ;MR2     "
015154  104007                                  ERROR   7                       ;MR3     "

015156  004437  023316                          JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

015162  004437  023740                          JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
015166  104010                                  ERROR   10                      ;MR1 CONTENTS IN ERROR
015170  104011                                  ERROR   11                      ;CS1 CONTENTS IN ERROR

015172  004437  024276                          JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
015176  104012                                  ERROR   12                      ;MR1 CONTENTS IN ERROR
```

```
015200  104013                            ERROR   13                   ;CS1 IN ERROR AFTER SEARCH
015202  104014                            ERROR   14                   ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

015204  004437  026500                    JSR     R4,DRGPSN            ;GO READ GAP AND SYNC
015210  104015                            ERROR   15                   ;MR1 IN ERROR READING GAP
015212  104016                            ERROR   16                   ;CS1 IN ERROR AFTER READING GAP
015214  104032                            ERROR   32                   ;MR1 IN ERROR READING SYNC
015216  104033                            ERROR   33                   ;CS1 IN ERROR AFTER READING SYNC

015220  012737  000001  002346            MOV     #1,ECPATX            ;LOAD EXPECTED PATTERN
015226  012737  005066  002350            MOV     #5066,ECPOSX         ;LOAD EXPECTED POSITION
015234  004437  027176                    JSR     R4,DREAD             ;GO SIMULATE DATA READ
015240  000400                            400                          ;NUMBER OF WORDS TO SIMULATE
015242  104034                            ERROR   34                   ;MR1 IN ERROR READING DATA OR ECC
015244  104035                            ERROR   35                   ;ECC ERROR READING DATA
015246  104036                            ERROR   36                   ;CS1 ERROR AFTER READING DATA OR ECC
015250  104041                            ERROR   41                   ;ECC REG INCORRECT AFTER ECC READ
015252  104042                            ERROR   42                   ;ERR IN ECC PAT CALC.
015254  104043                            ERROR   43                   ;ERR IN ECC POS COUNT


015256  012737  000022  002326            MOV     #22,BITCNT           ;SET BIT COUNT FOR CORRECTION COUNT
015264  012700  000005                    MOV     #5,R0                ;SET COUNT FOR ECCPOS PASS THRU 0
015270  012701  012672                    MOV     #12672,R1            ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
015274  005037  001214                    CLR     $TMP5                ;CLEAR SWITCH

015300  004737  031616          3$:       JSR     PC,RDBIT             ;GO READ BIT
015304  004737  030462                    JSR     PC,ECCGEN            ;GENERATE ECC
015310  016237  000032  002214            MOV     RKECPT(R2),T.ECPT           ;GET PATTERN
015316  016237  000030  002212            MOV     RKECPS(R2),T.ECPS           ;GET POSITION
015324  005237  002350                    INC     ECPOSX               ;BUMP POSITION EXPECTED
015330  013737  002350  002252            MOV     ECPOSX,E.ECPS        ;SET FOR COMPARE AND REPORT
015336  023737  002254  002214            CMP     E.ECPT,T.ECPT        ;CHECK IF PATTERN CORRECT
015344  001401                            BEQ     1$                   ;YES - SKIP
015346  104042                            ERROR   42
015350  023737  002252  002212  1$:       CMP     E.ECPS,T.ECPS        ;CHECK IF POSITION CORRECT
015356  001401                            BEQ     2$                   ;YES - SKIP
015360  104043                            ERROR   43
015362  005237  002326          2$:       INC     BITCNT               ;BUMP BIT COUNT
015366  005301                            DEC     R1                   ;DEC COUNT TO ZERO IN ECCPOS
015370  001343                            BNE     3$                   ;NOT YET ZERO - LOOP
015372  005300                            DEC     R0                   ;DEC PASS THRU 0 COUNT
015374  001406                            BEQ     4$                   ;IF 0 - SKIP
015376  012737  177777  002350            MOV     #-1,ECPOSX           ;PRESET EXPECTED POS TO GO TO ZERO
015404  012701  020000                    MOV     #20000,R1            ;SET COUNT TO NEXT 0 IN ECC POS
015410  000733                            BR      3$                   ;GO DO LOOP AGAIN

015412  005737  001214          4$:       TST     $TMP5                ;TEST SWITCH
015416  001012                            BNE     5$                   ;ALREADY SET - SKIP
015420  012701  010272                    MOV     #10272,R1            ;SET COUNT TO FINAL POSITION
015424  012700  000001                    MOV     #1,R0                ;SET R0 TO RETURN AFTER ONE PASS
015430  012737  177777  002350            MOV     #-1,ECPOSX           ;SET EXP POS TO GO TO 0 ON NEXT BIT
015436  005237  001214                    INC     $TMP5                ;BUMP SWITCH
015442  000716                            BR      3$                   ;GO DO LOOP AGAIN

015444  004737  031616          5$:       JSR     PC,RDBIT             ;ONE MORE BIT TO SET READY
```

```
015450  004437  032356                  JSR     R4,GETREG               ;GET '611 REGISTERS
015454  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
015462  052737  100200  002220          BIS     #RDY!CERR,E.CS1         ;SET READY AND ERROR
015470  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO
015476  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
015504  001402                          BEQ     6$                      ;YES - SKIP
015506  104044                          ERROR   44
015510  000410                          BR      7$
015512  032737  000100  002174  6$:     BIT     #ECH,T.ER               ;TEST IF ECC HARD ERROR SET
015520  001404                          BEQ     7$                      ;NO SKIP
015522  012737  100000  002234          MOV     #DCK,E.ER               ;SET EXPECTED ERROR REG
015530  104045                          ERROR   45

015532                          7$:

                                ;;*************************************************************
                                ;*TEST 24        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.
                                ;*
                                ;;*************************************************************
015532  000004                  TST24:  SCOPE
015534  012737  000001  001234          MOV     #1,$TIMES               ;;DO 1 ITERATION

015542  005037  002352                  CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
015546  012762  100000  000000          MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

015554  005037  002354                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
015560  004437  032230                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
015564  000000                          0                               ;CYLINDER 0
015566     000                          .BYTE   0                       ;SECTOR 0
015567     000                          .BYTE   0                       ;TRACK 0
015570  046632                          IBUFF                           ;BUFFER ADDRESS IBUFF
015572  177400                          -400                            ;WORD COUNT -400
015574  010021                          RDDATA!CFMT                             ;COMMAND RDDATA!CFMT

015576  004437  031770                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

015602  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
015606  000005                          5                               ;PATTERN 5


015610  012700  000734                  MOV     #734,R0                 ;SET INDEX INTO BUFFER
015614  012737  004020  050636          MOV     #4020,OBUFF+1000                ;STORE ECC WORDS
015622  012737  071720  050640          MOV     #71720,OBUFF+1002
015630  012737  000001  002344          MOV     #1,ECCSRC               ;SET SOURCE TO INDICATE ECC IN BUFFER
015636  042760  005000  047636          BIC     #BIT9!BIT11,OBUFF(R0)   ;CREATE 11 BIT BURST ERROR
015644  052760  050100  047636          BIS     #BIT6!BIT12!BIT14,OBUFF(R0)
```

```
015652  062700  000002                          ADD     #2,R0                   ;BUMP TO NEXT WORD
015656  052760  000001  047636                  BIS     #BIT0,OBUFF(R0)
015664  042760  000002  047636                  BIC     #BIT1,OBUFF(R0)         ;MAKE IT A 12 BIT BURST ERROR

015672  004437  032274                          JSR     R4,OPSTRT               ;START THE OPERATION
015676  004437  023334                          JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
015702  104004                                  ERROR   4                       ;CS1 MISCOMPARE
015704  104005                                  ERROR   5                       ;MR1     "
015706  104006                                  ERROR   6                       ;MR2     "
015710  104007                                  ERROR   7                       ;MR3     "

015712  004437  023316                          JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

015716  004437  023740                          JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
015722  104010                                  ERROR   10                      ;MR1 CONTENTS IN ERROR
015724  104011                                  ERROR   11                      ;CS1 CONTENTS IN ERROR

015726  004437  024276                          JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
015732  104012                                  ERROR   12                      ;MR1 CONTENTS IN ERROR
015734  104013                                  ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
015736  104014                                  ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

015740  004437  026500                          JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
015744  104015                                  ERROR   15                      ;MR1 IN ERROR READING GAP
015746  104016                                  ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
015750  104032                                  ERROR   32                      ;MR1 IN ERROR READING SYNC
015752  104033                                  ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC

015754  012737  000001  002346                  MOV     #1,ECPATX               ;LOAD EXPECTED PATTERN
015762  012737  005066  002350                  MOV     #5066,ECPOSX            ;LOAD EXPECTED POSITION
015770  004437  027176                          JSR     R4,DREAD                ;GO SIMULATE DATA READ
015774  000400                                  400                             ;NUMBER OF WORDS TO SIMULATE
015776  104034                                  ERROR   34                      ;MR1 IN ERROR READING DATA OR ECC
016000  104035                                  ERROR   35                      ;ECC ERROR READING DATA
016002  104036                                  ERROR   36                      ;CS1 ERROR AFTER READING DATA OR ECC
016004  104041                                  ERROR   41                      ;ECC REG INCORRECT AFTER ECC READ
016006  104042                                  ERROR   42                      ;ERR IN ECC PAT CALC.
016010  104043                                  ERROR   43                      ;ERR IN ECC POS COUNT


016012  012737  000022  002326                  MOV     #22,BITCNT              ;SET BIT COUNT FOR CORRECTION COUNT
016020  012700  000005                          MOV     #5,R0                   ;SET COUNT FOR ECCPOS PASS THRU 0
016024  012701  012672                          MOV     #12672,R1               ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
016030  005037  001214                          CLR     $TMPS                   ;CLEAR SWITCH

016034  004737  031616                  3$:     JSR     PC,RDBIT                ;GO READ BIT
016040  004737  030462                          JSR     PC,ECCGEN               ;GENERATE ECC
016044  016237  000032  002214                  MOV     RKECPT(R2),T.ECPT               ;GET PATTERN
016052  016237  000030  002212                  MOV     RKECPS(R2),T.ECPS               ;GET POSITION
016060  005237  002350                          INC     ECPOSX                  ;BUMP POSITION EXPECTED
016064  013737  002350  002252                  MOV     ECPOSX,E.ECPS           ;SET FOR COMPARE AND REPORT
016072  023737  002254  002214                  CMP     E.ECPT,T.ECPT           ;CHECK IF PATTERN CORRECT
016100  001401                                  BEQ     1$                      ;YES - SKIP
016102  104042                                  ERROR   42
016104  023737  002252  002212          1$:     CMP     E.ECPS,T.ECPS           ;CHECK IF POSITION CORRECT
016112  001401                                  BEQ     2$                      ;YES - SKIP
```

```
016114  104043                                  ERROR   43
016116  005237  002326                  2$:     INC     BITCNT                  ;BUMP BIT COUNT
016122  005301                                  DEC     R1                      ;DEC COUNT TO ZERO IN ECCPOS
016124  001343                                  BNE     3$                      ;NOT YET ZERO - LOOP
016126  005300                                  DEC     R0                      ;DEC PASS THRU 0 COUNT
016130  001406                                  BEQ     4$                      ;IF 0 - SKIP
016132  012737  177777  002350                  MOV     #-1,ECPOSX              ;PRESET EXPECTED POS TO GO TO ZERO
016140  012701  020000                          MOV     #20000,R1               ;SET COUNT TO NEXT 0 IN ECC POS
016144  000733                                  BR      3$                      ;GO DO LOOP AGAIN

016146  005737  001214                  4$:     TST     $TMP5                   ;TEST SWITCH
016152  001012                                  BNE     5$                      ;ALREADY SET - SKIP
016154  012701  011041                          MOV     #11041,R1               ;SET COUNT TO FINAL POSITION
016160  012700  000001                          MOV     #1,R0                   ;SET R0 TO RETURN AFTER ONE PASS
016164  012737  177777  002350                  MOV     #-1,ECPOSX              ;SET EXP POS TO GO TO 0 ON NEXT BIT
016172  005237  001214                          INC     $TMP5                   ;BUMP SWITCH
016176  000716                                  BR      3$                      ;GO DO LOOP AGAIN

016200  004737  031616                  5$:     JSR     PC,RDBIT                ;ONE MORE BIT TO SET READY
016204  004437  032356                          JSR     R4,GETREG               ;GET '611 REGISTERS
016210  013737  002260  002220                  MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
016216  052737  100200  002220                  BIS     #RDY!CERR,E.CS1         ;SET READY AND ERROR
016224  042737  000001  002220                  BIC     #GO,E.CS1               ;CLEAR GO
016232  023737  002160  002220                  CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
016240  001402                                  BEQ     6$                      ;YES - SKIP
016242  104044                                  ERROR   44
016244  000410                                  BR      7$
016246  032737  000100  002174          6$:     BIT     #ECH,T.ER               ;TEST IF ECC HARD ERROR SET
016254  001004                                  BNE     7$                      ;YES -  SKIP
016256  012737  100100  002234                  MOV     #DCK!ECH,E.ER           ;SET EXPECTED ERROR REG
016264  104045                                  ERROR   45

016266                                  7$:

                                        .SBTTL  **SPECIAL READ TESTS

                                        ;;*****************************************************
                                        ;*TEST 25         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.
                                        ;*
                                        ;;*****************************************************
016266  000004                          TST25:  SCOPE
016270  012737  000001  001234                  MOV     #1,$TIMES               ;;DO 1 ITERATION

016276  005037  002352                          CLR     BSEDES                  ;CLEAR BAD SEC DESIRED
016302  012762  100000  000000                  MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER

016310  005037  002344                          CLR     ECCSRC                  ;CLEAR ECC SOURCE
016314  005037  002354                          CLR     HIBITS                  ;CLEAR HI ORDER BITS
```

```
016320  004437  032230            JSR     R4,LOADRK           ;LOAD "L" REGISTERS
016324  000000                    0                           ;CYLINDER 0
016326     000                    .BYTE   0                   ;SECTOR 0
016327     000                    .BYTE   0                   ;TRACK 0
016330  046632                    IBUFF                       ;BUFFER ADDRESS IBUFF
016332  177675                    -103                        ;WORD COUNT -103
016334  000021                    RDDATA                      ;COMMAND RDDATA

016336  004437  031770            JSR     R4,CLRIBF           ;GO CLEAR INPUT BUFFER

016342  004437  032014            JSR     R4,BLDDAT           ;GO BUILD DATA
016346  000006                    6                           ;PATTERN 6


016350  004437  032274            JSR     R4,OPSTRT           ;START THE OPERATION
016354  004437  023334            JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
016360  104004                    ERROR   4                   ;CS1 MISCOMPARE
016362  104005                    ERROR   5                   ;MR1       "
016364  104006                    ERROR   6                   ;MR2       "
016366  104007                    ERROR   7                   ;MR3       "

016370  004437  023316            JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

016374  004437  023740            JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
016400  104010                    ERROR   10                  ;MR1 CONTENTS IN ERROR
016402  104011                    ERROR   11                  ;CS1 CONTENTS IN ERROR

016404  004437  024276            JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
016410  104012                    ERROR   12                  ;MR1 CONTENTS IN ERROR
016412  104013                    ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
016414  104014                    ERROR   14                  ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

016416  004437  026500            JSR     R4,DRGPSN           ;GO READ GAP AND SYNC
016422  104015                    ERROR   15                  ;MR1 IN ERROR READING GAP
016424  104016                    ERROR   16                  ;CS1 IN ERROR AFTER READING GAP
016426  104032                    ERROR   32                  ;MR1 IN ERROR READING SYNC
016430  104033                    ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

016432  012737  000000  002346    MOV     #0,ECPATX           ;LOAD EXPECTED PATTERN
016440  012737  004066  002350    MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
016446  004437  027176            JSR     R4,DREAD            ;GO SIMULATE DATA READ
016452  000400                    400                         ;NUMBER OF WORDS TO SIMULATE
016454  104034                    ERROR   34                  ;MR1 IN ERROR READING DATA OR ECC
016456  104035                    ERROR   35                  ;ECC ERROR READING DATA
016460  104036                    ERROR   36                  ;CS1 ERROR AFTER READING DATA OR ECC
016462  104041                    ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
016464  104042                    ERROR   42                  ;ERR IN ECC PAT CALC.
016466  104043                    ERROR   43                  ;ERR IN ECC POS COUNT


016470  013700  002164            MOV     T.BA,R0             ;GET ACTUAL BA
016474  162700  046632            SUB     #IBUFF,R0           ;SUB START VALUE OF BA
016500  022700  000206            CMP     #206,R0             ;TEST IF CORRECT NUM OF WORDS XFER
016504  001404                    BEQ     4$                  ;YES - SKIP
016506  012737  047040  002224    MOV     #IBUFF+206,E.BA     ;SET EXPECTED BA
016514  104046                    ERROR   46                  ;REPORT XFER LENGTH ERROR
```

```

016516  012701  000014          4$:     MOV     #3.*4.,R1               ;SET COUNT TO END OF OPERATION
016522  012762  000440  000026  3$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN THE CLOCK
016530  012762  000040  000026          MOV     #DMD,RKMR1(R2)
016536  005301                          DEC     R1                      ;DEC THE COUNT
016540  001370                          BNE     3$                      ;LOOP UNTIL ZERO

016542  004437  032356          1$:     JSR     R4,GETREG               ;GET 611 REGS
016546  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
016554  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO BIT
016562  052737  000200  002220          BIS     #RDY,E.CS1              ;SET READY BIT
016570  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
016576  001401                          BEQ     2$                      ;YES - SKIP
016600  104056                          ERROR   56                      ;"CS1 IN ERROR AFTER READ DATA"
016602                          2$:


                                ;;*****************************************************************
                                ;*TEST 26        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.
                                ;*
                                ;;*****************************************************************
016602  000004                  TST26:  SCOPE
016604  012737  000001  001234          MOV     #1,$TIMES               ;;DO 1 ITERATION

016612  012762  100000  000000          MOV     #CCLR,RKCS1(R2) ;CLEAR CONTROLLER

016620  005037  002352                  CLR     BSEDES                  ;CLEAR BSE DESIRED
016624  005037  002344                  CLR     ECCSRC                  ;CLEAR ECC SOURCE
016630  005037  002354                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
016634  004437  032230                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
016640  000000                          0                               ;CYLINDER 0
016642     000                          .BYTE   0                       ;SECTOR 0
016643     000                          .BYTE   0                       ;TRACK 0
016644  046632                          IBUFF                           ;BUFFER ADDRESS IBUFF
016646  177377                          -401                            ;WORD COUNT -401
016650  000021                          RDDATA                          ;COMMAND RDDATA

016652  004437  031770                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

016656  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
016662  000002                          2                               ;PATTERN 2


016664  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION
016670  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
016674  104004                          ERROR   4                       ;CS1 MISCOMPARE
```

```
016676  104005                          ERROR   5               ;MR1      "
016700  104006                          ERROR   6               ;MR2      "
016702  104007                          ERROR   7               ;MR3      "

016704  004437  023316                  JSR     R4,DSECTR       ;SIMULATE SECTOR PULSE

016710  004437  023740                  JSR     R4,DRSYNC       ;SIMULATE HEADER PREAMBLE
016714  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
016716  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR

016720  004437  024276                  JSR     R4,DHDCMP       ;SIMULATE HEADER SEARCH
016724  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
016726  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
016730  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

016732  004437  026500                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
016736  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
016740  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
016742  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
016744  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC

016746  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
016754  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
016762  004437  027176                  JSR     R4,DREAD        ;GO SIMULATE DATA READ
016766  000400                          400                     ;NUMBER OF WORDS TO SIMULATE
016770  104034                          ERROR   34              ;MR1 IN ERROR READING DATA OR ECC
016772  104035                          ERROR   35              ;ECC ERROR READING DATA
016774  104036                          ERROR   36              ;CS1 ERROR AFTER READING DATA OR ECC
016776  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
017000  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
017002  104043                          ERROR   43              ;ERR IN ECC POS COUNT


017004  012701  000040                  MOV     #8.*4.,R1       ;SET COUNT TO END OF SECTOR
017010  012762  000440  000026  6$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN THE CLOCK
017016  012762  000040  000026          MOV     #DMD,RKMR1(R2)
017024  005301                          DEC     R1              ;DEC THE COUNT
017026  001370                          BNE     6$              ;LOOP UNTIL ZERO

017030  052737  100000  002352          BIS     #BIT15,BSEDES   ;SET FOR BAD SECTOR ERROR

017036  004437  023316                  JSR     R4,DSECTR       ;ISSUE SECTOR PULSE

017042  004437  023740                  JSR     R4,DRSYNC       ;READ SYNC
017046  104010                          ERROR   10              ;MR1 CONTENTS IN ERROR
017050  104011                          ERROR   11              ;CS1 CONTENTS IN ERROR

017052  004437  024276                  JSR     R4,DHDCMP       ;DO HEADER COMPARE WITH BSE ERROR
017056  104012                          ERROR   12              ;MR1 CONTENTS IN ERROR
017060  104013                          ERROR   13              ;CS1 IN ERROR AFTER SEARCH
017062  000411                          BR      1$              ;EXPECTED RETURN (NO DA INCREMENT)

                                ;       IF THIS CODE IS EXECUTED, THE CONTROLLER DID NOT INHIBIT
                                ;       THE ADDRESS INCREMENT BECAUSE OF THE BSE ERROR.

017064  005037  002352                  CLR     BSEDES          ;CLEAR BSE DESIRED SWITCH
```

```
017070  013737  002274  002240          MOV     L.DCYL,E.DCYL     ;SET EXPECTED CYL
017076  013737  002266  002226          MOV     L.DA,E.DA         ;SET EXPECTED SEC/TRACK
017104  104047                          ERROR   47                ;UNEXPECTED ADDRESS INCREMENT

017106  016237  000014  002174  1$:     MOV     RKER(R2),T.ER     ;GET ERROR REG
017114  032737  000200  002174          BIT     #BSE,T.ER         ;TEST IF BSE SET
017122  001005                          BNE     2$                ;YES - SKIP
017124  012737  000200  002234          MOV     #BSE,E.ER         ;SET EXPECTED ERROR REG
017132  104051                          ERROR   51                ;"BSE DID NOT SET"
017134  000453                          BR      TST27             ;;GO TO NEXT TEST

017136  004437  032356          2$:     JSR     R4,GETREG         ;GET 611 REGS
017142  013737  002260  002220          MOV     L.CS1,E.CS1       ;SET EXPECTED CS1
017150  023737  002220  002160          CMP     E.CS1,T.CS1       ;CHECK IF CS1 OK
017156  001402                          BEQ     3$                ;YES - SKIP
017160  104056                          ERROR   56                ;"CS1 ERROR AFTER WRITE CHECK"
017162  000440                          BR      TST27             ;;GO TO NEXT TEST

017164  012701  010540          3$:     MOV     #278.*16.,R1      ;SET COUNT TO NEXT ECC FIELD
017170  005037  002320                  CLR     PR.BIT            ;CLEAR PRESENT AND PREVIOUS BITS
017174  005037  002322                  CLR     M1.BIT
017200  004737  031616          4$:     JSR     PC,RDBIT          ;SIMULATE READ BIT
017204  005301                          DEC     R1                ;DEC COUNT
017206  001374                          BNE     4$                ;LOOP UNTIL ZERO

017210  004437  032356                  JSR     R4,GETREG         ;GET 611 REGS
017214  042737  000001  002220          BIC     #GO,E.CS1         ;SET UP EXPECTED CS1
017222  052737  100200  002220          BIS     #RDY!CERR,E.CS1
017230  023737  002220  002160          CMP     E.CS1,T.CS1       ;CHECK IF CS1 OK
017236  001402                          BEQ     7$                ;YES - SKIP
017240  104056                          ERROR   56                ;"CS1 ERROR AFTER READ DATA"
017242  000410                          BR      TST27             ;;GO TO NEXT TEST

017244  012737  047632  002224  7$:     MOV     #IBUFF+1000,E.BA  ;SET EXPECTED BA
017252  023737  002224  002164          CMP     E.BA,T.BA         ;TEST IF BA CORRECT
017260  001401                          BEQ     5$                ;YES - SKIP
017262  104052                          ERROR   52                ;BUS ADDRESS INCORRECT

017264                          5$:

                                .SBTTL  **WRITE CHECK TESTS

                                ;;*********************************************************
                                ;*TEST 27       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.
                                ;*
                                ;;*********************************************************
017264  000004                  TST27:  SCOPE
017266  012737  000012  001234          MOV     #10.,$TIMES       ;;DO 10. ITERATIONS
```

```
017274  005037  002354                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
017300  005037  002352                  CLR     BSEDES                  ;CLEAR BSE DESIRED

017304  005037  002344                  CLR     ECCSRC                  ;CLEAR ECC SOURCE FLAG
017310  005037  002344                  CLR     ECCSRC                  ;CLEAR ECC SOURCE
017314  004437  032230                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
017320  000000                          0                               ;CYLINDER 0
017322     000                          .BYTE   0                       ;SECTOR 0
017323     000                          .BYTE   0                       ;TRACK 0
017324  046632                          IBUFF                           ;BUFFER ADDRESS IBUFF
017326  177400                          -400                            ;WORD COUNT -400
017330  000031                          WRTCHK                          ;COMMAND WRTCHK

017332  004437  031770                  JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER

017336  004437  032014                  JSR     R4,BLDDAT               ;GO BUILD DATA
017342  000004                          4                               ;PATTERN 4

017344  012700  000400                  MOV     #400,R0                 ;SET A COUNT FOR BUFFER SIZE
017350  012701  046632                  MOV     #IBUFF,R1               ;SET ADDRESS OF IBUFF
017354  012703  047636                  MOV     #OBUFF,R3               ;SET ADDRESS OF OBUFF
017360  012321                  75$:    MOV     (R3)+,(R1)+             ;MOV PATTERN TO IBUFF
017362  005300                          DEC     R0                      ;DEC COUNT
017364  001375                          BNE     75$                     ;LOOP UNTIL PATTERN IS MOVED

017366  012762  100000  000000  110$:   MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER
017374  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION
017400  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
017404  104004                          ERROR   4                       ;CS1 MISCOMPARE
017406  104005                          ERROR   5                       ;MR1        "
017410  104006                          ERROR   6                       ;MR2        "
017412  104007                          ERROR   7                       ;MR3        "

017414  004437  023316                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

017420  004437  023740                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
017424  104010                          ERROR   10                      ;MR1 CONTENTS IN ERROR
017426  104011                          ERROR   11                      ;CS1 CONTENTS IN ERROR

017430  004437  024276                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
017434  104012                          ERROR   12                      ;MR1 CONTENTS IN ERROR
017436  104013                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
017440  104014                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

017442  004437  026500                  JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
017446  104015                          ERROR   15                      ;MR1 IN ERROR READING GAP
017450  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
017452  104032                          ERROR   32                      ;MR1 IN ERROR READING SYNC
017454  104033                          ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC

017456  012737  000000  002346          MOV     #0,ECPATX               ;LOAD EXPECTED PATTERN
017464  012737  004066  002350          MOV     #4066,ECPOSX            ;LOAD EXPECTED POSITION
017472  004437  027176                  JSR     R4,DWRTCK               ;GO SIMULATE WRITE CHECK
017476  000400                          400                             ;NUMBER OF WORDS TO SIMULATE
017500  104053                          ERROR   53                      ;MR1 IN ERROR ON WRT CK OR ECC READ
017502  104054                          ERROR   54                      ;ECC ERROR IN WRITE CHECK OP
```

```
017504  104055                                 ERROR   55                 ;CS1 ERROR AFTER WRT CHK OR ECC READ
017506  104041                                 ERROR   41                 ;ECC REG INCORRECT AFTER ECC READ
017510  104042                                 ERROR   42                 ;ERR IN ECC PAT CALC.
017512  104043                                 ERROR   43                 ;ERR IN ECC POS COUNT


017514  012700  002710                         MOV     #40.*37.,R0        ;SET COUNT TO EMPTY SILO

017520  012762  000440  000026   1$:           MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
017526  012762  000040  000026                 MOV     #DMD,RKMR1(R2)
017534  005300                                 DEC     R0                 ;DEC COUNT
017536  001370                                 BNE     1$                 ;LOOP UNTIL 0

017540  004437  032356                         JSR     R4,GETREG          ;GET 611 REGS
017544  013737  002260  002220                 MOV     L.CS1,E.CS1        ;GET EXPECTED CS1
017552  042737  000001  002220                 BIC     #GO,E.CS1          ;CLEAR GO BIT
017560  052737  000200  002220                 BIS     #RDY,E.CS1         ;SET READY BIT
017566  023737  002160  002220                 CMP     T.CS1,E.CS1        ;CHECK IF CS1 CORRECT
017574  001401                                 BEQ     2$                 ;YES - SKIP
017576  104057                                 ERROR   57                 ;"CS1 IN ERROR AFTER WRITE CHECK"
017600                           2$:

                                 ;;*****************************************************************
                                 ;*TEST 30         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.
                                 ;*
                                 ;;*****************************************************************
017600  000004                   TST30:        SCOPE
017602  012737  000001  001234                 MOV     #1,$TIMES          ;;DO 1 ITERATION

017610  005037  002352                         CLR     BSEDES             ;CLEAR BAD SECTOR DESIRED
017614  005037  002354                         CLR     HIBITS             ;CLEAR HI ORDER BITS
017620  004437  032230                         JSR     R4,LOADRK          ;LOAD "L" REGISTERS
017624  000000                                 0                          ;CYLINDER 0
017626     000                                 .BYTE   0                  ;SECTOR 0
017627     000                                 .BYTE   0                  ;TRACK 0
017630  046632                                 IBUFF                      ;BUFFER ADDRESS IBUFF
017632  177400                                 -400                       ;WORD COUNT -400
017634  000031                                 WRTCHK                     ;COMMAND WRTCHK

017636  004437  031770                         JSR     R4,CLRIBF          ;GO CLEAR INPUT BUFFER

017642  004437  032014                         JSR     R4,BLDDAT          ;GO BUILD DATA
017646  000005                                 5                          ;PATTERN 5

017650  012700  000400                         MOV     #400,R0            ;SET A COUNT FOR BUFFER SIZE
```

```
017654  012701  046632                          MOV     #IBUFF,R1           ;SET ADDRESS OF IBUFF
017660  012703  047636                          MOV     #OBUFF,R3           ;SET ADDRESS OF OBUFF
017664  012321                          75$:    MOV     (R3)+,(R1)+         ;MOV PATTERN TO IBUFF
017666  005300                                  DEC     R0                  ;DEC COUNT
017670  001375                                  BNE     75$                 ;LOOP UNTIL PATTERN IS MOVED

017672  012700  001002                          MOV     #1002,R0            ;SET INDEX INTO BUFFER
017676  012737  126073  050636                  MOV     #126073,OBUFF+1000          ;STORE ECC WORDS
017704  012737  151052  050640                  MOV     #151052,OBUFF+1002
017712  012737  000001  002344                  MOV     #1,ECCSRC           ;SET SOURCE TO INDICATE ECC IN BUFFER
017720  032760  000100  047636                  BIT     #BIT6,OBUFF(R0)     ;TEST IF BIT SET
017726  001004                                  BNE     100$                ;YES - SKIP
017730  052760  000100  047636                  BIS     #BIT6,OBUFF(R0)     ;ELSE SET BIT
017736  000403                                  BR      101$
017740  042760  000100  047636          100$:   BIC     #BIT6,OBUFF(R0)     ;CLEAR BIT
017746                                  101$:

017746  012762  100000  000000          110$:   MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER
017754  004437  032274                          JSR     R4,OPSTRT           ;START THE OPERATION
017760  004437  023334                          JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
017764  104004                                  ERROR   4                   ;CS1 MISCOMPARE
017766  104005                                  ERROR   5                   ;MR1        "
017770  104006                                  ERROR   6                   ;MR2        "
017772  104007                                  ERROR   7                   ;MR3        "

017774  004437  023316                          JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

020000  004437  023740                          JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
020004  104010                                  ERROR   10                  ;MR1 CONTENTS IN ERROR
020006  104011                                  ERROR   11                  ;CS1 CONTENTS IN ERROR

020010  004437  024276                          JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
020014  104012                                  ERROR   12                  ;MR1 CONTENTS IN ERROR
020016  104013                                  ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
020020  104014                                  ERROR   14                  ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

020022  004437  026500                          JSR     R4,DRGPSN           ;GO READ GAP AND SYNC
020026  104015                                  ERROR   15                  ;MR1 IN ERROR READING GAP
020030  104016                                  ERROR   16                  ;CS1 IN ERROR AFTER READING GAP
020032  104032                                  ERROR   32                  ;MR1 IN ERROR READING SYNC
020034  104033                                  ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

020036  012737  000001  002346                  MOV     #1,ECPATX           ;LOAD EXPECTED PATTERN
020044  012737  004066  002350                  MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
020052  004437  027176                          JSR     R4,DWRTCK           ;GO SIMULATE WRITE CHECK
020056  000400                                  400                         ;NUMBER OF WORDS TO SIMULATE
020060  104053                                  ERROR   53                  ;MR1 IN ERROR ON WRT CK OR ECC READ
020062  104054                                  ERROR   54                  ;ECC ERROR IN WRITE CHECK OP
020064  104055                                  ERROR   55                  ;CS1 ERROR AFTER WRT CHK OR ECC READ
020066  104041                                  ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
020070  104042                                  ERROR   42                  ;ERR IN ECC PAT CALC.
020072  104043                                  ERROR   43                  ;ERR IN ECC POS COUNT


020074  012737  000020  002326                  MOV     #20,BITCNT          ;SET BIT COUNT FOR CORRECTION COUNT
020102  012700  000005                          MOV     #5,R0               ;SET COUNT FOR ECCPOS PASS THRU 0
```

```
020106  012701  013672                    MOV     #13672,R1           ;SET COUNT FOR ECCPOS TO 0 FIRST PASS
020112  005037  001214                    CLR     $TMP5               ;CLEAR SWITCH

020116  004737  031616           3$:      JSR     PC,RDBIT            ;GO READ BIT
020122  004737  030462                    JSR     PC,ECCGEN           ;GENERATE ECC
020126  016237  000032  002214            MOV     RKECPT(R2),T.ECPT         ;GET PATTERN
020134  016237  000030  002212            MOV     RKECPS(R2),T.ECPS         ;GET POSITION
020142  005237  002350                    INC     ECPOSX              ;BUMP POSITION EXPECTED
020146  013737  002350  002252            MOV     ECPOSX,E.ECPS       ;SET FOR COMPARE AND REPORT
020154  023737  002254  002214            CMP     E.ECPT,T.ECPT       ;CHECK IF PATTERN CORRECT
020162  001401                            BEQ     1$                  ;YES - SKIP
020164  104042                            ERROR   42
020166  023737  002252  002212   1$:      CMP     E.ECPS,T.ECPS       ;CHECK IF POSITION CORRECT
020174  001401                            BEQ     2$                  ;YES - SKIP
020176  104043                            ERROR   43
020200  005237  002326           2$:      INC     BITCNT              ;BUMP BIT COUNT
020204  005301                            DEC     R1                  ;DEC COUNT TO ZERO IN ECCPOS
020206  001343                            BNE     3$                  ;NOT YET ZERO - LOOP
020210  005300                            DEC     R0                  ;DEC PASS THRU 0 COUNT
020212  001406                            BEQ     4$                  ;IF 0 - SKIP
020214  012737  177777  002350            MOV     #-1,ECPOSX          ;PRESET EXPECTED POS TO GO TO ZERO
020222  012701  020000                    MOV     #20000,R1           ;SET COUNT TO NEXT 0 IN ECC POS
020226  000733                            BR      3$                  ;GO DO LOOP AGAIN

020230  005737  001214           4$:      TST     $TMP5               ;TEST SWITCH
020234  001012                            BNE     5$                  ;ALREADY SET - SKIP
020236  012701  010016                    MOV     #10016,R1           ;SET COUNT TO FINAL POSITION
020242  012700  000001                    MOV     #1,R0               ;SET R0 TO RETURN AFTER ONE PASS
020246  012737  177777  002350            MOV     #-1,ECPOSX          ;SET EXP POS TO GO TO 0 ON NEXT BIT
020254  005237  001214                    INC     $TMP5               ;BUMP SWITCH
020260  000716                            BR      3$                  ;GO DO LOOP AGAIN

020262  004737  031616           5$:      JSR     PC,RDBIT            ;ONE MORE BIT TO SET READY
020266  004437  032356                    JSR     R4,GETREG           ;GET '611 REGISTERS
020272  013737  002260  002220            MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
020300  052737  100200  002220            BIS     #RDY!CERR,E.CS1     ;SET READY AND ERROR
020306  042737  000001  002220            BIC     #GO,E.CS1           ;CLEAR GO
020314  023737  002160  002220            CMP     T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
020322  001402                            BEQ     6$                  ;YES - SKIP
020324  104044                            ERROR   44
020326  000410                            BR      7$
020330  032737  000100  002174   6$:      BIT     #ECH,T.ER           ;TEST IF ECC HARD ERROR SET
020336  001404                            BEQ     7$                  ;YES - SKIP
020340  012737  100000  002234            MOV     #DCK,E.ER           ;SET EXPECTED ERROR REG
020346  104045                            ERROR   45

020350                           7$:

                                 ;;*************************************************************
                                 ;*TEST 31        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 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.
                                        ;*
                                        ;;*******************************************************
020350  000004                          TST31:  SCOPE
020352  012737  000001  001234                  MOV     #1,$TIMES               ;;DO 1 ITERATION
020360  005037  002352                          CLR     BSEDES                  ;CLEAR BAD SECTOR DESIRED
020364  005037  002354                          CLR     HIBITS                  ;CLEAR HI ORDER BITS
020370  005037  002344                          CLR     ECCSRC                  ;CLEAR ECC SOURCE
020374  004437  032230                          JSR     R4,LOADRK               ;LOAD "L" REGISTERS
020400  000000                                  0                               ;CYLINDER 0
020402     000                                  .BYTE   0                       ;SECTOR 0
020403     000                                  .BYTE   0                       ;TRACK 0
020404  046632                                  IBUFF                           ;BUFFER ADDRESS IBUFF
020406  177400                                  -400                            ;WORD COUNT -400
020410  010031                                  WRTCHK!CFMT                             ;COMMAND WRTCHK!CFMT
020412  004437  031770                          JSR     R4,CLRIBF               ;GO CLEAR INPUT BUFFER
020416  004437  032014                          JSR     R4,BLDDAT               ;GO BUILD DATA
020422  000004                                  4                               ;PATTERN 4
020424  012700  000400                          MOV     #400,R0                 ;SET A COUNT FOR BUFFER SIZE
020430  012701  046632                          MOV     #IBUFF,R1               ;SET ADDRESS OF IBUFF
020434  012703  047636                          MOV     #OBUFF,R3               ;SET ADDRESS OF OBUFF
020440  012321                          75$:    MOV     (R3)+,(R1)+             ;MOV PATTERN TO IBUFF
020442  005300                                  DEC     R0                      ;DEC COUNT
020444  001375                                  BNE     75$                     ;LOOP UNTIL PATTERN IS MOVED
020446  012762  100000  000000          110$:   MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER
020454  004437  032274                          JSR     R4,OPSTRT               ;START THE OPERATION
020460  004437  023334                          JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
020464  104004                                  ERROR   4                       ;CS1 MISCOMPARE
020466  104005                                  ERROR   5                       ;MR1        "
020470  104006                                  ERROR   6                       ;MR2        "
020472  104007                                  ERROR   7                       ;MR3        "
020474  004437  023316                          JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE
020500  004437  023740                          JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
020504  104010                                  ERROR   10                      ;MR1 CONTENTS IN ERROR
020506  104011                                  ERROR   11                      ;CS1 CONTENTS IN ERROR
020510  004437  024276                          JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
020514  104012                                  ERROR   12                      ;MR1 CONTENTS IN ERROR
020516  104013                                  ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
020520  104014                                  ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH
020522  004437  026500                          JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
020526  104015                                  ERROR   15                      ;MR1 IN ERROR READING GAP
020530  104016                                  ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
020532  104032                                  ERROR   32                      ;MR1 IN ERROR READING SYNC
020534  104033                                  ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC
020536  012737  000000  002346                  MOV     #0,ECPATX               ;LOAD EXPECTED PATTERN
```

```
020544  012737  005066  002350          MOV     #5066,ECPOSX        ;LOAD EXPECTED POSITION
020552  004437  027176                  JSR     R4,DWRTCK           ;GO SIMULATE WRITE CHECK
020556  000400                          400                         ;NUMBER OF WORDS TO SIMULATE
020560  104053                          ERROR   53                  ;MR1 IN ERROR ON WRT CK OR ECC READ
020562  104054                          ERROR   54                  ;ECC ERROR IN WRITE CHECK OP
020564  104055                          ERROR   55                  ;CS1 ERROR AFTER WRT CHK OR ECC READ
020566  104041                          ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
020570  104042                          ERROR   42                  ;ERR IN ECC PAT CALC.
020572  104043                          ERROR   43                  ;ERR IN ECC POS COUNT
020574  012700  003100                  MOV     #40.*40.,R0         ;SET COUNT TO EMPTY SILO
020600  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;CLOCK CONTROLLER
020606  012762  000040  000026          MOV     #DMD,RKMR1(R2)
020614  005300                          DEC     R0                  ;DEC COUNT
020616  001370                          BNE     1$                  ;LOOP UNTIL 0
020620  004437  032356                  JSR     R4,GETREG           ;GET 611 REGS
020624  013737  002260  002220          MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
020632  042737  000001  002220          BIC     #GO,E.CS1           ;CLEAR GO BIT
020640  052737  000200  002220          BIS     #RDY,E.CS1          ;SET READY BIT
020646  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK IF CS1 CORRECT
020654  001401                          BEQ     2$                  ;YES - SKIP
020656  104057                          ERROR   57                  ;"CS1 IN ERROR AFTER WRITE CHECK"
020660                          2$:
                                ;*****************************************************************
                                ;*TEST 32          WRITE CHECK OF A PARTIAL SECTOR
                                ;*
                                ;*      CLEAR RK611 CONTROLLER WITH A CONTROLLER CLEAR.
                                ;*      PUT CONTROLLER IN DIAGNOSTIC MODE.  ISSUE A WRITE
                                ;*      CHECK OF 1 WORD TO AN RK06 IN 26 SECTOR FORMAT,
                                ;*      CYLINER 0, HEAD 0, SECTOR 0.  CLOCK THROUGH SEEK
                                ;*      AND DRIVE CLEAR MESSAGES.  SIMULATE A SECTOR PULSE,
                                ;*      A GOOD HEADER, TAKE ONE WORD TO BE WRITE-CHECKED, 377 WORDS
                                ;*      NOT TO BE WRITE CHECKED, AND A GOOD 32 BIT ECC.
                                ;*      MAKE SURE NO ERRORS SET.
                                ;*
                                ;*****************************************************************
020660  000004                  TST32:  SCOPE
020662  012737  000001  001234          MOV     #1,$TIMES           ;;DO 1 ITERATION
020670  005037  002352                  CLR     BSEDES              ;CLEAR BAD SECTOR DESIRED
020674  005037  002354                  CLR     HIBITS              ;CLEAR HI ORDER BITS
020700  005037  002344                  CLR     ECCSRC              ;CLEAR ECC SOURCE
020704  004437  032230                  JSR     R4,LOADRK           ;LOAD "L" REGISTERS
020710  000000                          0                           ;CYLINDER 0
020712     000                          .BYTE   0                   ;SECTOR 0
020713     000                          .BYTE   0                   ;TRACK 0
020714  046632                          IBUFF                       ;BUFFER ADDRESS IBUFF
020716  177777                          -1                          ;WORD COUNT -1
020720  000031                          WRTCHK                      ;COMMAND WRTCHK
020722  004437  031770                  JSR     R4,CLRIBF           ;GO CLEAR INPUT BUFFER
020726  004437  032014                  JSR     R4,BLDDAT           ;GO BUILD DATA
```

```
020732  000006                                  6                            ;PATTERN 6

020734  012700  000400                          MOV     #400,R0              ;SET A COUNT FOR BUFFER SIZE
020740  012701  046632                          MOV     #IBUFF,R1            ;SET ADDRESS OF IBUFF
020744  012703  047636                          MOV     #OBUFF,R3            ;SET ADDRESS OF OBUFF
020750  012321                          75$:    MOV     (R3)+,(R1)+          ;MOV PATTERN TO IBUFF
020752  005300                                  DEC     R0                   ;DEC COUNT
020754  001375                                  BNE     75$                  ;LOOP UNTIL PATTERN IS MOVED

020756  012762  100000  000000          110$:   MOV     #CCLR,RKCS1(R2)      ;CLEAR CONTROLLER
020764  004437  032274                          JSR     R4,OPSTRT            ;START THE OPERATION
020770  004437  023334                          JSR     R4,DISEEK            ;SIMULATE IMPLIED SEEK
020774  104004                                  ERROR   4                    ;CS1 MISCOMPARE
020776  104005                                  ERROR   5                    ;MR1      "
021000  104006                                  ERROR   6                    ;MR2      "
021002  104007                                  ERROR   7                    ;MR3      "

021004  004437  023316                          JSR     R4,DSECTR            ;SIMULATE SECTOR PULSE

021010  004437  023740                          JSR     R4,DRSYNC            ;SIMULATE HEADER PREAMBLE
021014  104010                                  ERROR   10                   ;MR1 CONTENTS IN ERROR
021016  104011                                  ERROR   11                   ;CS1 CONTENTS IN ERROR

021020  004437  024276                          JSR     R4,DHDCMP            ;SIMULATE HEADER SEARCH
021024  104012                                  ERROR   12                   ;MR1 CONTENTS IN ERROR
021026  104013                                  ERROR   13                   ;CS1 IN ERROR AFTER SEARCH
021030  104014                                  ERROR   14                   ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

021032  004437  026500                          JSR     R4,DRGPSN            ;GO READ GAP AND SYNC
021036  104015                                  ERROR   15                   ;MR1 IN ERROR READING GAP
021040  104016                                  ERROR   16                   ;CS1 IN ERROR AFTER READING GAP
021042  104032                                  ERROR   32                   ;MR1 IN ERROR READING SYNC
021044  104033                                  ERROR   33                   ;CS1 IN ERROR AFTER READING SYNC

021046  012737  000000  002346                  MOV     #0,ECPATX            ;LOAD EXPECTED PATTERN
021054  012737  004066  002350                  MOV     #4066,ECPOSX         ;LOAD EXPECTED POSITION
021062  004437  027176                          JSR     R4,DWRTCK            ;GO SIMULATE WRITE CHECK
021066  000400                                  400                          ;NUMBER OF WORDS TO SIMULATE
021070  104053                                  ERROR   53                   ;MR1 IN ERROR ON WRT CK OR ECC READ
021072  104054                                  ERROR   54                   ;ECC ERROR IN WRITE CHECK OP
021074  104055                                  ERROR   55                   ;CS1 ERROR AFTER WRT CHK OR ECC READ
021076  104041                                  ERROR   41                   ;ECC REG INCORRECT AFTER ECC READ
021100  104042                                  ERROR   42                   ;ERR IN ECC PAT CALC.
021102  104043                                  ERROR   43                   ;ERR IN ECC POS COUNT


021104  013700  002164                          MOV     T.BA,R0              ;GET ACTUAL BA
021110  162700  046632                          SUB     #IBUFF,R0            ;SUB START VALUE OF BA
021114  022700  000002                          CMP     #2,R0                ;TEST IF CORRECT NUM OF WORDS XFER
021120  001404                                  BEQ     4$                   ;YES - SKIP
021122  012737  046634  002224                  MOV     #IBUFF+2,E.BA        ;SET EXPECTED BA
021130  104046                                  ERROR   46                   ;REPORT XFER LENGTH ERROR

021132  012701  000040                  4$:     MOV     #8.*4.,R1            ;SET COUNT TO END OF OPERATION
021136  012762  000440  000026          3$:     MOV     #DMD!MCLK,RKMR1(R2)  ;RUN CLOCK
021144  012762  000040  000026                  MOV     #DMD,RKMR1(R2)
```

```
021152  005301                          DEC     R1                      ;DEC COUNT
021154  001370                          BNE     3$                      ;LOOP UNTIL ZERO

021156  004437  032356          1$:     JSR     R4,GETREG               ;GET 611 REGS
021162  013737  002260  002220          MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
021170  042737  000001  002220          BIC     #GO,E.CS1               ;CLEAR GO BIT
021176  052737  000200  002220          BIS     #RDY,E.CS1              ;SET READY BIT
021204  023737  002160  002220          CMP     T.CS1,E.CS1             ;CHECK IF CS1 CORRECT
021212  001401                          BEQ     2$                      ;YES - SKIP
021214  104057                          ERROR   57                      ;"CS1 IN ERROR AFTER WRITE CHECK"
021216                          2$:


                                ;;*****************************************************************
                                ;*TEST 33        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
                                ;*
                                ;;*****************************************************************
021216  000004                  TST33:  SCOPE
021220  012737  000001  001234          MOV     #1,$TIMES               ;;DO 1 ITERATION

021226  005037  002352                  CLR     BSEDES                  ;CLEAR BAD SECTOR DESIRED
021232  005037  002354                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
021236  005037  002344                  CLR     ECCSRC                  ;CLEAR ECC SOURCE
021242  004437  032230                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
021246  000000                          0                               ;CYLINDER 0
021250     000                          .BYTE   0                       ;SECTOR 0
021251     000                          .BYTE   0                       ;TRACK 0
021252  046632                          IBUFF                           ;BUFFER ADDRESS IBUFF
021254  177777                          -1                              ;WORD COUNT -1
021256  000031                          WRTCHK                          ;COMMAND WRTCHK

021260  012705  000020                  MOV     #16.,R5                 ;SET PATTERN COUNT
021264  005037  047636                  CLR     OBUFF                   ;CLEAR FIRST WORD OF OBUFF
021270  012737  000001  046632          MOV     #1,IBUFF                ;SET IN FIRST PATTERN
021276  032737  000001  046632          BIT     #BIT0,IBUFF             ;IS BIT 0 SET
021304  001003                          BNE     112$                    ;YES - PATTERN SHOULD BE 0
021306  012737  177777  047636          MOV     #-1,OBUFF               ;ELSE PATTERN SHOULD BE ALL ONES
021314  012737  021322  001110  112$:   MOV     #110$,$LPERR            ;SET LOCAL LOOP ON ERROR

021322  012762  100000  000000  110$:   MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER
021330  004437  032274                  JSR     R4,OPSTRT               ;START THE OPERATION
```

```
021334  004437  023334                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
021340  104004                          ERROR   4                       ;CS1 MISCOMPARE
021342  104005                          ERROR   5                       ;MR1        "
021344  104006                          ERROR   6                       ;MR2        "
021346  104007                          ERROR   7                       ;MR3        "

021350  004437  023316                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

021354  004437  023740                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
021360  104010                          ERROR   10                      ;MR1 CONTENTS IN ERROR
021362  104011                          ERROR   11                      ;CS1 CONTENTS IN ERROR

021364  004437  024276                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
021370  104012                          ERROR   12                      ;MR1 CONTENTS IN ERROR
021372  104013                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
021374  104014                          ERROR   14                      ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

021376  004437  026500                  JSR     R4,DRGPSN               ;GO READ GAP AND SYNC
021402  104015                          ERROR   15                      ;MR1 IN ERROR READING GAP
021404  104016                          ERROR   16                      ;CS1 IN ERROR AFTER READING GAP
021406  104032                          ERROR   32                      ;MR1 IN ERROR READING SYNC
021410  104033                          ERROR   33                      ;CS1 IN ERROR AFTER READING SYNC

021412  012737  000000  002346          MOV     #0,ECPATX               ;LOAD EXPECTED PATTERN
021420  012737  004066  002350          MOV     #4066,ECPOSX            ;LOAD EXPECTED POSITION
021426  004437  027176                  JSR     R4,DWRTCK               ;GO SIMULATE WRITE CHECK
021432  000001                          1                               ;NUMBER OF WORDS TO SIMULATE
021434  104053                          ERROR   53                      ;MR1 IN ERROR ON WRT CK OR ECC READ
021436  104054                          ERROR   54                      ;ECC ERROR IN WRITE CHECK OP
021440  104055                          ERROR   55                      ;CS1 ERROR AFTER WRT CHK OR ECC READ
021442  104041                          ERROR   41                      ;ECC REG INCORRECT AFTER ECC READ
021444  104042                          ERROR   42                      ;ERR IN ECC PAT CALC.
021446  104043                          ERROR   43                      ;ERR IN ECC POS COUNT

021450  012701  000040                  MOV     #8.*4.,R1               ;SET COUNT
021454  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
021462  012762  000040  000026          MOV     #DMD,RKMR1(R2)
021470  005301                          DEC     R1                      ;DEC COUNT
021472  001370                          BNE     1$                      ;LOOP UNTIL ZERO

021474  004437  032356                  JSR     R4,GETREG               ;GET REGS
021500  032737  040000  002170          BIT     #WCE,T.CS2              ;TEST IF WCE SET
021506  001012                          BNE     2$                      ;YES - SKIP
021510  013737  002170  001162          MOV     T.CS2,$REG0             ;SET CS2 FOR REPORT
021516  013737  046632  001164          MOV     IBUFF,$REG1             ;SET WORD READ
021524  013737  047636  001166          MOV     OBUFF,$REG2             ;SET WORD FROM BUFFER
021532  104060                          ERROR   60                      ;"WCE FAILED TO SET"

021534  104415                  2$:     SCOP1                           ;LOCAL LOOP ON ERROR

021536  006337  046632                  ASL     IBUFF                   ;SHIFT FOR NEXT TEST PATTERN
021542  005305                          DEC     R5                      ;DEC PATTERN COUNT
021544  001266                          BNE     110$                    ;LOOP UNTIL ALL PATTERNS TESTED

                                ;;*************************************************************
                                ;*TEST 34        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
                                                ;*
                                                ;;*******************************************************
021546  000004                                  TST34:  SCOPE
021550  012737  000001  001234                          MOV     #1,$TIMES               ;;DO 1 ITERATION

021556  005037  002352                                  CLR     BSEDES                  ;CLEAR BAD SECTOR DESIRED
021562  005037  002354                                  CLR     HIBITS                  ;CLEAR HI ORDER BITS
021566  005037  002344                                  CLR     ECCSRC                  ;CLEAR ECC SOURCE
021572  004437  032230                                  JSR     R4,LOADRK               ;LOAD "L" REGISTERS
021576  000000                                          0                               ;CYLINDER 0
021600     000                                          .BYTE   0                       ;SECTOR 0
021601     000                                          .BYTE   0                       ;TRACK 0
021602  046632                                          IBUFF                           ;BUFFER ADDRESS IBUFF
021604  177777                                          -1                              ;WORD COUNT -1
021606  000031                                          WRTCHK                          ;COMMAND WRTCHK

021610  012705  000020                                  MOV     #16.,R5                 ;SET PATTERN COUNT
021614  005037  047636                                  CLR     OBUFF                   ;CLEAR FIRST WORD OF OBUFF
021620  012737  177776  046632                          MOV     #177776,IBUFF           ;SET IN FIRST PATTERN
021626  032737  000001  046632                          BIT     #BIT0,IBUFF             ;IS BIT 0 SET
021634  001003                                          BNE     112$                    ;YES - PATTERN SHOULD BE 0
021636  012737  177777  047636                          MOV     #-1,OBUFF               ;ELSE PATTERN SHOULD BE ALL ONES
021644  012737  021652  001110  112$:                   MOV     #110$,$LPERR            ;SET LOCAL LOOP ON ERROR

021652  012762  100000  000000  110$:                   MOV     #CCLR,RKCS1(R2)         ;CLEAR CONTROLLER
021660  004437  032274                                  JSR     R4,OPSTRT               ;START THE OPERATION
021664  004437  023334                                  JSR     R4,DISEEK               ;SIMULATE IMPLIED SEEK
021670  104004                                          ERROR   4                       ;CS1 MISCOMPARE
021672  104005                                          ERROR   5                       ;MR1     "
021674  104006                                          ERROR   6                       ;MR2     "
021676  104007                                          ERROR   7                       ;MR3     "

021700  004437  023316                                  JSR     R4,DSECTR               ;SIMULATE SECTOR PULSE

021704  004437  023740                                  JSR     R4,DRSYNC               ;SIMULATE HEADER PREAMBLE
021710  104010                                          ERROR   10                      ;MR1 CONTENTS IN ERROR
021712  104011                                          ERROR   11                      ;CS1 CONTENTS IN ERROR

021714  004437  024276                                  JSR     R4,DHDCMP               ;SIMULATE HEADER SEARCH
021720  104012                                          ERROR   12                      ;MR1 CONTENTS IN ERROR
021722  104013                                          ERROR   13                      ;CS1 IN ERROR AFTER SEARCH
```

```
021724  104014                          ERROR   14              ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

021726  004437  026500                  JSR     R4,DRGPSN       ;GO READ GAP AND SYNC
021732  104015                          ERROR   15              ;MR1 IN ERROR READING GAP
021734  104016                          ERROR   16              ;CS1 IN ERROR AFTER READING GAP
021736  104032                          ERROR   32              ;MR1 IN ERROR READING SYNC
021740  104033                          ERROR   33              ;CS1 IN ERROR AFTER READING SYNC

021742  012737  000000  002346          MOV     #0,ECPATX       ;LOAD EXPECTED PATTERN
021750  012737  004066  002350          MOV     #4066,ECPOSX    ;LOAD EXPECTED POSITION
021756  004437  027176                  JSR     R4,DWRTCK       ;GO SIMULATE WRITE CHECK
021762  000001                          1                       ;NUMBER OF WORDS TO SIMULATE
021764  104053                          ERROR   53              ;MR1 IN ERROR ON WRT CK OR ECC READ
021766  104054                          ERROR   54              ;ECC ERROR IN WRITE CHECK OP
021770  104055                          ERROR   55              ;CS1 ERROR AFTER WRT CHK OR ECC READ
021772  104041                          ERROR   41              ;ECC REG INCORRECT AFTER ECC READ
021774  104042                          ERROR   42              ;ERR IN ECC PAT CALC.
021776  104043                          ERROR   43              ;ERR IN ECC POS COUNT

022000  012701  000040                  MOV     #8.*4.,R1       ;SET COUNT
022004  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
022012  012762  000040  000026          MOV     #DMD,RKMR1(R2)
022020  005301                          DEC     R1              ;DEC COUNT
022022  001370                          BNE     1$              ;LOOP UNTIL ZERO

022024  004437  032356                  JSR     R4,GETREG       ;GET REGS
022030  032737  040000  002170          BIT     #WCE,T.CS2      ;TEST IF WCE SET
022036  001012                          BNE     2$              ;YES - SKIP
022040  013737  002170  001162          MOV     T.CS2,$REG0     ;SET CS2 FOR REPORT
022046  013737  046632  001164          MOV     IBUFF,$REG1     ;SET WORD READ
022054  013737  047636  001166          MOV     OBUFF,$REG2     ;SET WORD FROM BUFFER
022062  104060                          ERROR   60              ;"WCE FAILED TO SET"

022064  104415                  2$:     SCOP1                   ;LOCAL LOOP ON ERROR

022066  006337  046632                  ASL     IBUFF           ;SHIFT FOR NEXT TEST PATTERN
022072  005537  046632                  ADC     IBUFF           ;PUT CARRY BACK IN
022076  005305                          DEC     R5              ;DEC PATTERN COUNT
022100  001264                          BNE     110$            ;LOOP UNTIL ALL PATTERNS TESTED

                                ;;*************************************************************
                                ;*TEST 35        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.
                                ;*
                                ;;*************************************************************
022102  000004                  TST35:  SCOPE
022104  012737  000001  001234          MOV     #1,$TIMES       ;;DO 1 ITERATION
```

```

022112  005037  002352                       CLR     BSEDES              ;CLEAR BAD SECTOR DESIRED
022116  005037  002344                       CLR     ECCSRC              ;CLEAR ECC SOURCE
022122  004437  032230                       JSR     R4,LOADRK           ;LOAD "L" REGISTERS
022126  000000                               0                           ;CYLINDER 0
022130     000                               .BYTE   0                   ;SECTOR 0
022131     000                               .BYTE   0                   ;TRACK 0
022132  046632                               IBUFF                       ;BUFFER ADDRESS IBUFF
022134  177777                               -1                          ;WORD COUNT -1
022136  010031                               WRTCHK!CFMT                         ;COMMAND WRTCHK!CFMT

022140  012737  040000  002354               MOV     #40000,HIBITS       ;SET HIBITS FOR PATTERN OF 200000
022146  005037  046632                       CLR     IBUFF               ;SET LOW BITS FOR READ
022152  005037  047636                       CLR     OBUFF               ;SET 0 IN BUFFER
022156  012737  022164  001110      112$:    MOV     #110$,$LPERR        ;SET LOCAL LOOP ON ERROR
022164  012762  100000  000000      110$:    MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER

022172  012700  004000                       MOV     #4000,R0            ;SET STALL COUNT
022176  005300                      4$:      DEC     R0                  ;DEC COUNT
022200  001376                               BNE     4$                  ;LOOP UNTIL ZERO

022202  004437  032274                       JSR     R4,OPSTRT           ;START THE OPERATION
022206  004437  023334                       JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
022212  104004                               ERROR   4                   ;CS1 MISCOMPARE
022214  104005                               ERROR   5                   ;MR1       "
022216  104006                               ERROR   6                   ;MR2       "
022220  104007                               ERROR   7                   ;MR3       "

022222  004437  023316                       JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

022226  004437  023740                       JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
022232  104010                               ERROR   10                  ;MR1 CONTENTS IN ERROR
022234  104011                               ERROR   11                  ;CS1 CONTENTS IN ERROR

022236  004437  024276                       JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
022242  104012                               ERROR   12                  ;MR1 CONTENTS IN ERROR
022244  104013                               ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
022246  104014                               ERROR   14                  ;RKDCYL OR RKDA IN ERROR OFTER SEARCH

022250  004437  026500                       JSR     R4,DRGPSN           ;GO READ GAP AND SYNC
022254  104015                               ERROR   15                  ;MR1 IN ERROR READING GAP
022256  104016                               ERROR   16                  ;CS1 IN ERROR AFTER READING GAP
022260  104032                               ERROR   32                  ;MR1 IN ERROR READING SYNC
022262  104033                               ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

022264  012737  000000  002346               MOV     #0,ECPATX           ;LOAD EXPECTED PATTERN
022272  012737  005066  002350               MOV     #5066,ECPOSX        ;LOAD EXPECTED POSITION
022300  004437  027176                       JSR     R4,DWRTCK           ;GO SIMULATE WRITE CHECK
022304  000001                               1                           ;NUMBER OF WORDS TO SIMULATE
022306  104053                               ERROR   53                  ;MR1 IN ERROR ON WRT CK OR ECC READ
022310  104054                               ERROR   54                  ;ECC ERROR IN WRITE CHECK OP
022312  104055                               ERROR   55                  ;CS1 ERROR AFTER WRT CHK OR ECC READ
022314  104041                               ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
022316  104042                               ERROR   42                  ;ERR IN ECC PAT CALC.
022320  104043                               ERROR   43                  ;ERR IN ECC POS COUNT

```

```
022322  012701  000040                          MOV     #8.*4.,R1          ;SET COUNT
022326  012762  000440  000026          1$:     MOV     #DMD!MCLK,RKMR1(R2) ;RUN CLOCK
022334  012762  000040  000026                  MOV     #DMD,RKMR1(R2)
022342  005301                                  DEC     R1                 ;DEC COUNT
022344  001370                                  BNE     1$                 ;LOOP UNTIL ZERO

022346  004437  032356                          JSR     R4,GETREG          ;GET REGS
022352  032737  040000  002170                  BIT     #WCE,T.CS2         ;TEST IF WCE SET
022360  001025                                  BNE     2$                 ;YES - SKIP
022362  013737  002170  001162                  MOV     T.CS2,$REG0        ;SET CS2 FOR REPORT
022370  013737  046632  001164                  MOV     IBUFF,$REG1        ;SET WORD READ
022376  013737  047636  001166                  MOV     OBUFF,$REG2        ;SET WORD FROM BUFFER
022404  005037  001170                          CLR     $REG3              ;SET UP FOR REPORT
022410  013737  002354  001172                  MOV     HIBITS,$REG4
022416  012700  000016                          MOV     #14.,R0
022422  006037  001172                  3$:     ROR     $REG4
022426  005300                                  DEC     R0
022430  001374                                  BNE     3$
022432  104063                                  ERROR   63                 ;"WCE FAILED TO SET"

022434  104415                          2$:     SCOP1                      ;LOCAL LOOP ON ERROR

022436  023727  002354  100000                  CMP     HIBITS,#100000     ;TEST IF SECOND PATTERN DONE
022444  001404                                  BEQ     5$                 ;YES - EXIT
022446  012737  100000  002354                  MOV     #100000,HIBITS     ;SET FOR PATTERN 400000
022454  000643                                  BR      110$               ;LOOP UNTIL ALL PATTERNS TESTED
022456                                  5$:

                                        ;;*****************************************************************
                                        ;*TEST 36        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.
                                        ;*
                                        ;;*****************************************************************
022456  000004                          TST36:  SCOPE
022460  012737  000001  001234                  MOV     #1,$TIMES          ;;DO 1 ITERATION

022466  005037  002354                          CLR     HIBITS             ;CLEAR HI BITS
022472  005037  002352                          CLR     BSEDES             ;CLEAR BAD SECTOR DESIRED
022476  004437  032230                          JSR     R4,LOADRK          ;LOAD "L" REGISTERS
022502  000000                                  0                          ;CYLINDER 0
022504     000                                  .BYTE   0                  ;SECTOR 0
022505     000                                  .BYTE   0                  ;TRACK 0
022506  046632                                  IBUFF                      ;BUFFER ADDRESS IBUFF
022510  177400                                  -400                       ;WORD COUNT -400
022512  000031                                  WRTCHK                     ;COMMAND WRTCHK

022514  004437  031770                          JSR     R4,CLRIBF          ;GO CLEAR INPUT BUFFER
```

```
022520  004437  032014                  JSR     R4,BLDDAT           ;GO BUILD DATA
022524  000005                          5                           ;PATTERN 5

022526  012700  000400                  MOV     #400,R0             ;SET A COUNT FOR BUFFER SIZE
022532  012701  046632                  MOV     #IBUFF,R1           ;SET ADDRESS OF IBUFF
022536  012703  047636                  MOV     #OBUFF,R3           ;SET ADDRESS OF OBUFF
022542  012321                  75$:    MOV     (R3)+,(R1)+         ;MOV PATTERN TO IBUFF
022544  005300                          DEC     R0                  ;DEC COUNT
022546  001375                          BNE     75$                 ;LOOP UNTIL PATTERN IS MOVED
022550  012700  000530                  MOV     #530,R0             ;SET INDEX INTO BUFFER

022554  012737  126073  050636          MOV     #126073,OBUFF+1000          ;STORE ECC WORDS
022562  012737  151052  050640          MOV     #151052,OBUFF+1002
022570  012737  000001  002344          MOV     #1,ECCSRC           ;SET SOURCE TO INDICATE ECC IN BUFFER
022576  032760  000100  047636          BIT     #BIT6,OBUFF(R0)     ;TEST IF BIT SET
022604  001004                          BNE     100$                ;YES - SKIP
022606  052760  000100  047636          BIS     #BIT6,OBUFF(R0)     ;ELSE SET BIT
022614  000403                          BR      101$
022616  042760  000100  047636  100$:   BIC     #BIT6,OBUFF(R0)     ;CLEAR BIT
022624                          101$:

022624  012762  100000  000000  110$:   MOV     #CCLR,RKCS1(R2)     ;CLEAR CONTROLLER
022632  004437  032274                  JSR     R4,OPSTRT           ;START THE OPERATION
022636  004437  023334                  JSR     R4,DISEEK           ;SIMULATE IMPLIED SEEK
022642  104004                          ERROR   4                   ;CS1 MISCOMPARE
022644  104005                          ERROR   5                   ;MR1     "
022646  104006                          ERROR   6                   ;MR2     "
022650  104007                          ERROR   7                   ;MR3     "

022652  004437  023316                  JSR     R4,DSECTR           ;SIMULATE SECTOR PULSE

022656  004437  023740                  JSR     R4,DRSYNC           ;SIMULATE HEADER PREAMBLE
022662  104010                          ERROR   10                  ;MR1 CONTENTS IN ERROR
022664  104011                          ERROR   11                  ;CS1 CONTENTS IN ERROR

022666  004437  024276                  JSR     R4,DHDCMP           ;SIMULATE HEADER SEARCH
022672  104012                          ERROR   12                  ;MR1 CONTENTS IN ERROR
022674  104013                          ERROR   13                  ;CS1 IN ERROR AFTER SEARCH
022676  104014                          ERROR   14                  ;RKDCYL OR RKDA IN ERROR AFTER SEARCH

022700  004437  026500                  JSR     R4,DRGPSN           ;GO READ GAP AND SYNC
022704  104015                          ERROR   15                  ;MR1 IN ERROR READING GAP
022706  104016                          ERROR   16                  ;CS1 IN ERROR AFTER READING GAP
022710  104032                          ERROR   32                  ;MR1 IN ERROR READING SYNC
022712  104033                          ERROR   33                  ;CS1 IN ERROR AFTER READING SYNC

022714  012737  000000  002346          MOV     #0,ECPATX           ;LOAD EXPECTED PATTERN
022722  012737  004066  002350          MOV     #4066,ECPOSX        ;LOAD EXPECTED POSITION
022730  004437  027176                  JSR     R4,DWRTCK           ;GO SIMULATE WRITE CHECK
022734  000310                          310                         ;NUMBER OF WORDS TO SIMULATE
022736  104053                          ERROR   53                  ;MR1 IN ERROR ON WRT CK OR ECC READ
022740  104054                          ERROR   54                  ;ECC ERROR IN WRITE CHECK OP
022742  104055                          ERROR   55                  ;CS1 ERROR AFTER WRT CHK OR ECC READ
022744  104041                          ERROR   41                  ;ECC REG INCORRECT AFTER ECC READ
022746  104042                          ERROR   42                  ;ERR IN ECC PAT CALC.
022750  104043                          ERROR   43                  ;ERR IN ECC POS COUNT
```

```


022752  012701  000020                  MOV     #4.*4.,R1           ;SET COUNT TO END OF OPERATION
022756  012762  000440  000026  1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;RUN CLOCK
022764  012762  000040  000026          MOV     #DMD,RKMR1(R2)
022772  005301                          DEC     R1                  ;DEC COUNT
022774  001370                          BNE     1$                  ;LOOP UNTIL ZERO

022776  004437  032356                  JSR     R4,GETREG           ;GET '611 REGS
023002  032737  040000  002170          BIT     #WCE,T.CS2          ;TEST IF WRITE CHECK SET
023010  001013                          BNE     2$                  ;YES - SKIP
023012  013737  002170  001162          MOV     T.CS2,$REG0         ;SET UP FOR REPORT
023020  016037  046632  001164          MOV     IBUFF(R0),$REG1     ;BUFFER WORD
023026  016037  047636  001166          MOV     OBUFF(R0),$REG2     ;SIMULATED WORD
023034  104060                          ERROR   60                  ;WCE NOT SET ERROR
023036  000423                          BR      4$                  ;EXIT

023040  012737  046634  002224  2$:     MOV     #IBUFF+2,E.BA       ;SET UP EXPECTED BA
023046  060037  002224                  ADD     R0,E.BA             ;ADD IN THE NUMBER OF WORDS TO THE ERROR
023052  023737  002224  002164          CMP     E.BA,T.BA           ;CHECK IF CORRECT
023060  001402                          BEQ     3$                  ;YES - SKIP
023062  104061                          ERROR   61                  ;BUS ADDRESS ERROR
023064  000410                          BR      4$                  ;EXIT

023066  012737  177655  002222  3$:     MOV     #-123,E.WC          ;SET EXPECTED WORD COUNT
023074  023737  002162  002222          CMP     T.WC,E.WC           ;CHECK IF CORRECT
023102  001401                          BEQ     4$                  ;YES - SKIP
023104  104062                          ERROR   62                  ;WORD COUNT ERROR

023106                          4$:

```

```
                                        .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

023106                                  $EOP:
023106  000004                                  SCOPE
023110  005037  001102                          CLR     $TSTNM                  ;;ZERO THE TEST NUMBER
023114  005037  001234                          CLR     $TIMES                  ;;ZERO THE NUMBER OF ITERATIONS
023120  005237  001256                          INC     $PASS                   ;;INCREMENT THE PASS NUMBER
023124  042737  100000  001256                  BIC     #100000,$PASS           ;;DON'T ALLOW A NEG. NUMBER
023132  005327                                  DEC     (PC)+                   ;;LOOP?
023134  000001                          $EOPCT: .WORD   1
023136  003063                                  BGT     $DOAGN                  ;;YES
023140  012737                                  MOV     (PC)+,@(PC)+            ;;RESTORE COUNTER
023142  000001                          $ENDCT: .WORD   1
023144  023134                                          $EOPCT
023146  104401  023154                          TYPE    ,65$                    ;;TYPE ASCIZ STRING
023152  000407                                  BR      64$                     ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <12><15>/END PASS #/
023172                                  64$:
023172  013746  001256                          MOV     $PASS,-(SP)             ;;SAVE $PASS FOR TYPEOUT
                                                                                ;;TYPE PASS NUMBER
023176  104405                                  TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
023200  104401  023206                          TYPE    ,67$                    ;;TYPE ASCIZ STRING
023204  000421                                  BR      66$                     ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
023250                                  66$:
023250  013746  001112                          MOV     $ERTTL,-(SP)            ;;SAVE $ERTTL FOR TYPEOUT
                                                                                ;;TOTAL NUMBER OF ERRORS
023254  104405                                  TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
023256  104401  001245                          TYPE    ,$CRLF                  ;;TYPE CARRIAGE RETURN, LINE FEED
023262  005037  001112                          CLR     $ERTTL                  ;;CLEAR ERROR TOTAL
023266  013700  000042                  $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
023272  001405                                  BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
023274  000005                                  RESET                           ;;CLEAR THE WORLD
023276  004710                          $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
023300  000240                                  NOP                             ;;SAVE ROOM
023302  000240                                  NOP                             ;;FOR
023304  000240                                  NOP                             ;;ACT11
023306                                  $DOAGN:
023306  000137                                  JMP     @(PC)+                  ;;RETURN
023310  003364                          $RTNAD: .WORD   NEWPAS
023312     377     377     000          $ENULL: .BYTE   -1,-1,0                 ;;NULL CHARACTER STRING
        023316                                  .EVEN

                                        .SBTTL  DIAGNOSTIC MODE SECTOR PULSE
                                        ;*      THE SECTOR PULSE IS PROVIDED BY THIS ROUTINE
023316  012762  000140  000026          DSECTR: MOV     #DMD!MSP,RKMR1(R2)      ;SET SECTOR PULE
023324  012762  000040  000026                  MOV     #DMD,RKMR1(R2)  ;RESET SECTOR PULSE
023332  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

 023334                                 DISEEK:
 023334 010046                                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
 023336 010346                                  MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 023340 032737  000002  002260                  BIT     #BIT1,L.CS1             ;CHECK IF WRITE DATA
 023346 001003                                  BNE     10$                     ;YES - SKIP
 023350 012700  000316                          MOV     #51.*4+2,R0             ;SET COUNT FOR READ OR WRITE CHECK
 023354 000412                                  BR      1$

 023356 012700  004612                  10$:    MOV     #66.*37.,R0             ;COUNT FOR CLOCK THROUGH SILO LOAD
 023362 032737  010000  002260                  BIT     #CFMT,L.CS1             ;TEST IF 22 SECTOR MODE
 023370 001402                                  BEQ     11$                     ;NO - SKIP
 023372 012700  005222                          MOV     #66.*41.,R0             ;ELSE CHANGE FOR 18 BIT WORDS
 023376 062700  000016                  11$:    ADD     #4*3+2,R0               ;AND SHIFT REGISTER LOAD

 023402 012762  000440  000026          1$:     MOV     #DMD!MCLK,RKMR1(R2)     ;SET CLOCK
 023410 012762  000040  000026                  MOV     #DMD,RKMR1(R2)          ;CLEAR CLOCK
 023416 005300                                  DEC     R0                      ;LOOP UNTIL R0 IS ZERO
 023420 001370                                  BNE     1$

 023422 013703  002274                          MOV     L.DCYL,R3               ;GET DESIRED CYLINDER
 023426 004437  032444                          JSR     R4,FSBLVV               ;REVERSE ORDER OF BITS
 023432 010337  002246                          MOV     R3,E.MR2                ;STORE AS EXPECTED MR2
 023436 113703  002267                          MOVB    L.DT,R3                 ;GET DESIRED TRACK/SECTOR
 023442 042703  177400                          BIC     #177400,R3              ;CLEAR UNUSED BITS
 023446 012700  000005                          MOV     #5,R0                   ;SET SHIFT COUNT
 023452 006303                          15$:    ASL     R3                      ;SHIFT FOR TRACK ALIGNMENT
 023454 005300                                  DEC     R0                      ;DEC COUNT
 023456 001375                                  BNE     15$                     ;LOOP UNTIL ZERO
 023460 153703  002266                          BISB    L.DS,R3                 ;INSERT SECTOR NUMBER
 023464 032737  010000  002260                  BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
 023472 001402                                  BEQ     16$                     ;NO - SKIP
 023474 052703  001000                          BIS     #BIT9,R3                ;ELSE SET FORMAT BIT IN HDR WRD
 023500 004437  032444                  16$:    JSR     R4,FSBLVV               ;REVERSE BIT ORDER
 023504 010337  002250                          MOV     R3,E.MR3                ;STORE AS EXPECTED MR3

 023510 013737  002260  002220                  MOV     L.CS1,E.CS1             ;GET EXPECTED CS1
 023516 012737  022040  002244                  MOV     #DMD!ECCW!MEWD,E.MR1    ;SET UP EXPECTED MR1
 023524 004437  032356                          JSR     R4,GETREG               ;GET RK REGS
```

```

023530  023737  002160  002220          CMP     T.CS1,E.CS1         ;CHECK CS1
023536  001411                          BEQ     2$                  ; OK - SKIP
023540  012737  023562  001236          MOV     #2$,$ESCAPE         ;SET ESCAPE TO RETURN FOR
                                                                    ;REMAINING TESTS
023546  032777  001000  155364          BIT     #BIT9,@SWR          ;CHECK IF LOOP ON ERROR
023554  001056                          BNE     9$                  ;YES - SKIP

023556  010446                  4$:     MOV     R4,-(SP)            ;SET STACK FOR RETURN TO SUB-ROUTINE
023560  000204                          RTS     R4

023562  005724                  2$:     TST     (R4)+               ;BUMP TO NEXT ERROR CALL
023564  023737  002244  002204          CMP     E.MR1,T.MR1         ;CHECK MR1
023572  001410                          BEQ     3$                  ;OK - SKIP
023574  032777  001000  155336          BIT     #BIT9,@SWR          ;CHECK IF LOOP ON ERROR
023602  001043                          BNE     9$                  ;YES - SKIP
023604  012737  023614  001236          MOV     #3$,$ESCAPE         ;SET ESCAPE FOR RETURN TO SUB-ROUTINE
023612  000761                          BR      4$

023614  005724                  3$:     TST     (R4)+               ;SAME OPERATION AS ABOVE TO
023616  023737  002246  002206          CMP     E.MR2,T.MR2         ;CHECK MR2
023624  001410                          BEQ     5$
023626  032777  001000  155304          BIT     #BIT9,@SWR
023634  001026                          BNE     9$
023636  012737  023646  001236          MOV     #5$,$ESCAPE
023644  000744                          BR      4$

023646  005724                  5$:     TST     (R4)+               ;SAME OPERATION AS ABOVE TO
023650  023737  002250  002210          CMP     E.MR3,T.MR3         ;CHECK MR3
023656  001414                          BEQ     8$
023660  032777  001000  155252          BIT     #BIT9,@SWR
023666  001011                          BNE     9$
023670  012737  023700  001236          MOV     #6$,$ESCAPE
023676  000727                          BR      4$

023700  032777  001000  155232  6$:     BIT     #BIT9,@SWR          ;CHECK IF LOOP ON ERROR
023706  001001                          BNE     9$                  ;YES - SKIP
023710  005724                  8$:     TST     (R4)+               ;BUMP TO NO ERROR RETURN
023712  005037  001236          9$:     CLR     $ESCAPE             ;CLEAR ESCAPE
023716  012603                          MOV     (SP)+,R3            ;;POP STACK INTO R3
023720  012600                          MOV     (SP)+,R0            ;;POP STACK INTO R0
023722  000204                          RTS     R4

023724  005037  001236          7$:     CLR     $ESCAPE             ;CLEAR ESCAPE
023730  012706  001100                  MOV     #STACK,SP           ;CLEAN OFF STACK
023734  000177  155150                  JMP     @$LPERR             ;GO TO LOOP START

                                .SBTTL  DIAGNOSTIC MODE READ SYNC PATTERN (PREAMBLE)
                                ;*      THE 255 "0" BITS AND 1 "1" BIT ARE CLOCKED THROUTH THE READ.
                                ;*      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 ABORT
                                        ;*      THE PRESENT TEST UNLESS LOOP ON ERROR IS SET.

        023740                          DRSYNC:
        023740  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
        023742  005037  002326                  CLR     BITCNT                  ;CLEAR BIT COUNT
        023746  005037  001236                  CLR     $ESCAPE                 ;CLEAR ESCAPE
        023752  012700  000200                  MOV     #128.,R0                ;SET COUNT TO CLOCK TO READ GATE
        023756  005037  002320                  CLR     PR.BIT                  ;CLEAR PRESENT BIT
        023762  005037  002322                  CLR     M1.BIT                  ;CLEAR PREVIOUS BIT
        023766  012737  022040  002244          MOV     #DMD!MEWD!ECCW,E.MR1 ;SET UP EXPECTED MR1

        023774  004737  031616          3$:     JSR     PC,RDBIT                ;GO READ A BIT
        024000  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;GET MR1
        024006  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF IT IS CORRECT
        024014  001424                          BEQ     2$                      ;YES - SKIP

        024016  012737  024066  001236          MOV     #2$,$ESCAPE             ;SET ESCAPE FOR CONTINUE TEST

        024024  005037  001236          4$:     CLR     $ESCAPE                 ;CLEAR ESCAPE
        024030  032777  001000  155102          BIT     #BIT9,@SWR              ;CHECK IF LOOP ON ERROR
        024036  001011                          BNE     1$                      ;YES SKIP
        024040  013700  001254                  MOV     $TESTN,R0               ;ELSE SET UP TO SKIP TO NEXT TEST
        024044  006300                          ASL     R0
        024046  016037  033472  001236          MOV     $SW08TB(R0),$ESCAPE
        024054  162737  000002  001236          SUB     #2,$ESCAPE
        024062                          1$:
        024062  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
        024064  000204                          RTS     R4

        024066  005237  002326          2$:     INC     BITCNT                  ;INCREMENT THE BIT COUNT
        024072  005300                          DEC     R0                      ;LOOP UNTIL READY FOR READ GATE
        024074  001337                          BNE     3$

        024076  012700  000177                  MOV     #127.,R0                ;SET COUNT FOR REST OF SYNC
        024102  012737  122040  002244          MOV     #DMD!MEWD!ECCW!RDGATE,E.MR1 ;SET NEW EXPECTED MR1

        024110  004737  031616          5$:     JSR     PC,RDBIT                ;GO READ BIT
        024114  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;GET MR1
        024122  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF CORRECT
        024130  001404                          BEQ     6$                      ;YES - SKIP
        024132  012737  024142  001236          MOV     #6$,$ESCAPE             ;SET FOR CONTINUE TESTING
        024140  000445                          BR      50$                     ;GO TO ERROR EXIT
        024142  005237  002326          6$:     INC     BITCNT                  ;BUMP BIT COUNTER
        024146  005300                          DEC     R0                      ;LOOP UNTIL READY FOR SYNC 1
        024150  001357                          BNE     5$

        024152  013737  002320  002322          MOV     PR.BIT,M1.BIT           ;MOV PRESENT TO PREVIOUS
        024160  012737  000001  002320          MOV     #1,PR.BIT               ;SET SYNC 1 BIT
        024166  004737  031616                  JSR     PC,RDBIT                ;GO READ BIT
        024172  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;GET MR1
        024200  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF CORRECT
        024206  001404                          BEQ     7$                      ;YES - SKIP
```

```
024210  012737  024220  001236             MOV     #7$,$ESCAPE         ;SET RETURN FOR CONTINUE TEST
024216  000416                             BR      50$                 ;GO TO ERROR EXIT

024220  004437  032356          7$:        JSR     R4,GETREG           ;GO GET RK611 REGISTERS
024224  005237  002326                     INC     BITCNT              ;BUMP BIT COUNTER
024230  005724                             TST     (R4)+               ;BUMP TO NEXT ERROR CALL
024232  013737  002260  002220             MOV     L.CS1,E.CS1         ;GET EXPECTED CS1
024240  023737  002160  002220             CMP     T.CS1,E.CS1         ;CHECK IF CORRECT
024246  001266                             BNE     4$                  ;NO - SKIP

024250  005724                             TST     (R4)+               ;BUMP TO NO ERROR RETURN
024252  000703                             BR      1$                  ;SKIP TO RETURN

024254  032777  001000  154656  50$:       BIT     #BIT9,@SWR          ;TEST IF LOOP ON ERROR
024262  001403                             BEQ     51$                 ;NO - SKIP
024264  005037  001236                     CLR     $ESCAPE             ;CLEAR FOR SKIP TO NEXT TEST
024270  000674                             BR      1$
024272  010446                  51$:       MOV     R4,-(SP)            ;SET UP STACK FOR RETURN
024274  000204                             RTS     R4

                                .SBTTL  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, TRACK,
                                ;*      AND SECTOR SPECIFIED IN THE "L" REGISTERS. THE READING OF THE
                                ;*      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 IS
                                ;*      VERIFIED. IF THE INCREMENT IS GOOD, THE "L" REGISTERS ARE
                                ;*      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 ERROR.
                                ;*
                                ;*      CALL:
                                ;*      JSR     R4,DHDCMP
                                ;*      ERROR   12      ;ERROR IN MR1
                                ;*      ERROR   13      ;ERROR IN CS1
                                ;*      ERROR   14      ;ERROR IN PACK ADDRESS

024276                          DHDCMP:
024276  010046                             MOV     R0,-(SP)            ;;PUSH R0 ON STACK
024300  010146                             MOV     R1,-(SP)            ;;PUSH R1 ON STACK.
024302  010346                             MOV     R3,-(SP)            ;;PUSH R3 ON STACK
024304  010546                             MOV     R5,-(SP)            ;;PUSH R5 ON STACK
024306  005037  002326                     CLR     BITCNT              ;CLEAR BIT COUNTER
024312  012700  000005                     MOV     #5,R0               ;SET A SHIFT COUNT
024316  113701  002267                     MOVB    L.DT,R1             ;GET DESIRED TRACK
024322  042701  177400                     BIC     #177400,R1          ;CLEAR UNUSED BITS

024326  006301                  1$:        ASL     R1                  ;SHIFT TRACK TO ALIGN TO
024330  005300                             DEC     R0                  ;HEADER WORD FORMAT
024332  001375                             BNE     1$
```

```
024334  153701  002266                   BISB    L.DS,R1                 ;INSERT DESIRED SECTOR
024340  052701  140000                   BIS     #BIT15!BIT14,R1         ;SET BS BITS
024344  013700  002274                   MOV     L.DCYL,R0               ;GET DESIRED CYLINDER
024350  032737  010000  002260           BIT     #CFMT,L.CS1             ;TEST IF 22 SECTOR FORMAT
024356  001402                           BEQ     2$                      ;NO - SKIP
024360  052701  001000                   BIS     #BIT9,R1                ;ELSE SET FORMAT BIT
024364  010003                   2$:     MOV     R0,R3                   ;COMPUTE VRC
024366  010105                           MOV     R1,R5
024370  040005                           BIC     R0,R5
024372  040103                           BIC     R1,R3
024374  050503                           BIS     R5,R3
024376  005737  002352                   TST     BSEDES                  ;TEST IF BSE DESIRED
024402  001404                           BEQ     23$                     ;NO - SKIP
024404  043701  002352                   BIC     BSEDES,R1               ;CLEAR BIT IN WORD 2
024410  043703  002352                   BIC     BSEDES,R3               ;CORRECT HVRC
024414                           23$:

                                 ;       R0, R1, AND R3 HAVE THE EXPECTED HEADERS

024414  005037  001204                   CLR     $TMP1                   ;CLEAR A PASS COUNTER
024420  012737  122040  002244           MOV     #DMD!MEWD!ECCW!RDGATE,E.MR1  ;SET EXPECTED MR1

024426  012705  000001           11$:    MOV     #BIT0,R5                ;ESTABLISH BEGINNING OF DATA WORD
024432  013737  002320  002322   7$:     MOV     PR.BIT,M1.BIT           ;STORE PREVIOUS BIT
024440  005037  002320                   CLR     PR.BIT                  ;PRESET FOR PRESENT BIT 0
024444  030500                           BIT     R5,R0                   ;TEST IF BIT TO BE A ONE
024446  001403                           BEQ     3$                      ;NO - SKIP
024450  012737  000001  002320           MOV     #BIT0,PR.BIT            ;ELSE CHANGE TO ONE
024456  004737  031616           3$:     JSR     PC,RDBIT                ;GO READ A BIT
024462  016237  000026  002204           MOV     RKMR1(R2),T.MR1         ;GET MR1 FOR TESTING
024470  023737  002204  002244           CMP     T.MR1,E.MR1             ;CHECK IF CORRECT
024476  001431                           BEQ     6$                      ;YES - SKIP
024500  012737  024562  001236           MOV     #6$,$ESCAPE             ;SET RETURN TO CONTINUE TEST
024506  000137  025160                   JMP     50$

024512  005037  001236           4$:     CLR     $ESCAPE                 ;CLEAR TO JUMP TO NEXT TEST
024516  032777  001000  154414           BIT     #BIT9,@SWR              ;CHECK IF LOOP ON ERROR
024524  001011                           BNE     5$                      ;YES - SKIP
024526  013705  001254                   MOV     $TESTN,R5               ;GET CURRENT TEST NUMBER
024532  006305                           ASL     R5                      ;AND SET UP TO ESCAPE TO NEXT TEST
024534  016537  033472  001236           MOV     $SW08TB(R5),$ESCAPE
024542  162737  000002  001236           SUB     #2,$ESCAPE

024550                           5$:
024550  012605                           MOV     (SP)+,R5                ;;POP STACK INTO R5
024552  012603                           MOV     (SP)+,R3                ;;POP STACK INTO R3
024554  012601                           MOV     (SP)+,R1                ;;POP STACK INTO R1
024556  012600                           MOV     (SP)+,R0                ;;POP STACK INTO R0
024560  000204                           RTS     R4

024562  005237  002326           6$:     INC     BITCNT                  ;BUMP BIT COUNT
024566  005705                           TST     R5                      ;CHECK IF LAST BIT OF THIS WORD
024570  100402                           BMI     8$                      ;HAS BEEN SIMULATED - YES - SKIP
024572  006305                           ASL     R5                      ;SHIFT TO NEXT BIT
024574  000716                           BR      7$                      ;GO DO THIS BIT

```

```
024576  005737  001204          8$:     TST     $TMP1           ;TEST IF PASS FOR 1ST WORD JUST DONE
024602  001004                          BNE     9$              ;NO - SKIP
024604  010100                          MOV     R1,R0           ;PUT WORD 2 IN R0 FOR SIMULATION
024606  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
024612  000705                          BR      11$

024614  023727  001204  000002  9$:     CMP     $TMP1,#2        ;JUST DONE WITH WORD 2?
024622  002004                          BGE     10$             ;NO SKIP
024624  010300                          MOV     R3,R0           ;PUT 3RD WORD IN R0
024626  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
024632  000675                          BR      11$

024634  023727  001204  000003  10$:    CMP     $TMP1,#3        ;JUST DONE WITH WORD 3?
024642  002006                          BGE     16$             ;NO - SKIP
024644  005000                          CLR     R0              ;PUT IN WORD 0 OF GAP
024646  012705  100000                  MOV     #BIT15,R5       ;SET TO GIVE IT 1ST GAP BIT
024652  005237  001204                  INC     $TMP1           ;BUMP PASS COUNT
024656  000665                          BR      7$

024660  023727  001204  000004  16$:    CMP     $TMP1,#4        ;TEST IF READY FOR GAP
024666  001436                          BEQ     12$             ;YES - SKIP

                                ;       THE FOLLOWING BLOCK OF CODE COMPUTES THE NUMBER OF CLOCK
                                ;       PULSES (WHICH ARE ACTUALLY CALLS TO RDBIT) THE OPERATION
                                ;       BEING PERFORMED REQUIRES TO UPDATE THE ADDRESS COUNTERS
                                ;       (RKDC AND RKDA). THIS CALCULATION IS BASED ON THE WHAT WAS
                                ;       IN THESE COUNTERS WHEN THE OPERATION STARTED (L.DC AND L.DA).

024670  012701  000025                  MOV     #25,R1          ;PRESET R1 FOR 24 SECTOR MODE
024674  032737  010000  002260          BIT     #CFMT,L.CS1     ;IS IT 24 SECTOR MODE OPERATION?
024702  001402                          BEQ     21$             ;YES - SKIP
024704  012701  000023                  MOV     #23,R1          ;ELSE CHANGE FOR 22 SECTOR MODE
024710  052701  001000          21$:    BIS     #BIT9,R1        ;SET FOR MAX HEAD OF 2
024714  023701  002266                  CMP     L.DS,R1         ;TEST IF TRACK/SECTOR ADDRESS WAS MAX
024720  001003                          BNE     17$             ;NO - SKIP
024722  012705  000100                  MOV     #BIT6,R5        ;SET FOR 8 BITS OF GAP TO UPDATE
024726  000410                          BR      19$
024730  123701  002266          17$:    CMPB    L.DS,R1         ;TEST IF SECTOR WAS MAX
024734  001003                          BNE     18$             ;NO - SKIP
024736  012705  000400                  MOV     #BIT8,R5        ;SET FOR 6 BITS OF GAP TO UPDATE
024742  000402                          BR      19$
024744  012705  010000          18$:    MOV     #BIT12,R5       ;SET TO GIVE 1ST 4 BITS OF GAP
024750  005237  001204          19$:    INC     $TMP1           ;BUMP PASS COUNT
024754  042737  100000  002244          BIC     #RDGATE,E.MR1   ;CLEAR READ GATE
024762  000623                          BR      7$
                                ;       END OF BLOCK

024764  004437  032356          12$:    JSR     R4,GETREG       ;GET RK611 REGISTERS
024770  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
024772  013737  002260  002220          MOV     L.CS1,E.CS1     ;SET EXPECTED CS1
025000  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
025006  001241                          BNE     4$              ;NO SKIP

025010  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
025012  013737  002274  002240          MOV     L.DCYL,E.DCYL   ;GET STARTING CYLINDER
025020  013737  002266  002226          MOV     L.DA,E.DA       ;               TRACK/SECTOR
```

```
025026  105237  002226                       INCB    E.DA            ;BUMP SECTOR
025032  012700  000026                       MOV     #26,R0          ;PRESET FOR 26 SEC/TRACK
025036  032737  010000  002260               BIT     #CFMT,L.CS1     ;TEST IF IN 26 SECTOR MODE
025044  001402                               BEQ     13$             ;YES - SKIP
025046  012700  000024                       MOV     #24,R0          ;ELSE CHANGE FOR 24 SECTOR MODE
025052  120037  002226               13$:    CMPB    R0,E.DA         ;CHECK IF SECTOR TO LARGE
025056  001015                               BNE     15$             ;NO - SKIP
025060  105037  002226                       CLRB    E.DA            ;SET SECTOR TO 0
025064  105237  002227                       INCB    E.DA+1          ;BUMP TRACK
025070  012700  000003                       MOV     #3,R0           ;PRESET FOR 3 TRACKS (RK06)
025074  123700  002227                       CMPB    E.DA+1,R0       ;CHECK IF TRACK INC TO LARGE
025100  001004                               BNE     15$             ;NO - SKIP

025102  105037  0C2227                       CLRB    E.DA+1          ;CLEAR TRACK TO 0
025106  005237  )02240                       INC     E.DCYL          ;BUMP CYLINDER

025112  023737  002240  002200       15$:    CMP     E.DCYL,T.DCYL   ;CHECK IF CYL OK
025120  001004                               BNE     25$             ;NO - SKIP
025122  023737  002226  002166               CMP     E.DA,T.DA       ;CHECK IF DA OK
025130  001402                               BEQ     26$
025132  000137  024512               25$:    JMP     4$

025136  013737  002240  002274       26$:    MOV     E.DCYL,L.DCYL   ;STORE COMPUTED NEXT CYL
025144  013737  002226  002266               MOV     E.DA,L.DA       ;AND NEXT DA

025152  005724                               TST     (R4)+           ;BUMP TO NO ERROR RETURN
025154  000137  024550                       JMP     5$              ;GO TO GOOD EXIT

025160  032777  001000  153752       50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
025166  001404                               BEQ     51$             ;NO - SKIP
025170  005037  001236                       CLR     $ESCAPE         ;CLEAR TO LOOP ON ERROR
025174  000137  024550                       JMP     5$

025200  010446                       51$:    MOV     R4,-(SP)        ;SET STACK TO CONTINUE TEST
025202  000204                               RTS     R4
                                     .SBTTL  DIAGNOSTIC MODE GAP READ AND SYNC WRITE
                                     ;*      THE READING OF THE GAP AND WRITING OF THE SYNC (OR PREAMBLE)
                                     ;*      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
```

```
025204                                  DWGPSN:
025204  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.
025206  005737  002266                          TST     L.DA                    ;TEST IF DESIRED TRACK/SECTOR NOW 0
025212  001006                                  BNE     8$                      ;NO - SKIP
025214  012700  000066                          MOV     #54.,R0                 ;ELSE ADJUST COUNT FOR CYL CHANGE
025220  012737  000011  002326                  MOV     #11,BITCNT              ;PRESET BIT COUNT FOR REST OF GAP
025226  000416                                  BR      10$
025230  105737  002266                  8$:     TSTB    L.DS                    ;TEST IF NEW SECTOR IS 0
025234  001006                                  BNE     9$                      ;NO - SKIP
025236  012700  000070                          MOV     #56.,R0                 ;ELSE ADJUST COUNT FOR TRACK CHANGE
025242  012737  000007  002326                  MOV     #7,BITCNT               ;PRESET BIT COUNT FOR REST OF GAP
025250  000405                                  BR      10$
025252  012737  000005  002326          9$:     MOV     #5,BITCNT               ;PRESET BIT COUNTER FOR REST OF GAP
025260  012700  000074                          MOV     #60.,R0                 ;SET COUNT FOR 60 BITS
                                        ;       END OF BLOCK

025264  012737  022040  002244          10$:    MOV     #DMD!ECCW!MEWD,E.MR1    ;SET EXPECTED MR1
025272  005037  002320                          CLR     PR.BIT                  ;CLEAR PRESENT BIT
025276  005037  002322                          CLR     M1.BIT                  ;CLEAR PREVIOUS BIT
025302  004737  031616                  1$:     JSR     PC,RDBIT                ;GO READ BIT
025306  016237  000026  002204                  MOV     RKMR1(R2),T.MR1         ;GET MR1
025314  023737  002244  002204                  CMP     E.MR1,T.MR1             ;CHECK IF CORRECT
025322  001404                                  BEQ     11$                     ;YES - SKIP
025324  012737  025346  001236                  MOV     #2$,$ESCAPE             ;SET RETURN TO CONT TEST
025332  000513                                  BR      50$
025334  005237  002326                  11$:    INC     BITCNT                  ;BUMP BIT COUNT
025340  005300                                  DEC     R0                      ;DEC LOOP COUNT
025342  001357                                  BNE     1$                      ;LOOP IF NOT ZERO
025344  000421                                  BR      4$                      ;ELSE SKIP

025346  005037  001236                  2$:     CLR     $ESCAPE                 ;CLEAR TO EXIT OR LOOP
025352  022777  001000  153560                  CMP     #BIT9,@SWR              ;ERROR RETURN: CHECK IF LOOP ON ERROR
025360  001011                                  BNE     3$                      ;YES - SKIP
025362  013700  001254                          MOV     $TESTN,R0               ;GET NUMBER OF NEXT TEST
025366  006300                                  ASL     R0                      ;SHIFT FOR INDEXING
025370  016037  033472  001236                  MOV     $SW08TB(R0),$ESCAPE     ;SET ESCAPE FOR GOT TO NEXT TEST
025376  162737  000002  001236                  SUB     #2,$ESCAPE              ;SET TO FIRST INST (SCOPE)
025404                                  3$:
025404  012600                                  MOV     (SP)+,R0                ;;POP STACK INTO R0
025406  000204                                  RTS     R4

025410  004437  032356                  4$:     JSR     R4,GETREG               ;GET RK611 REGS
025414  005724                                  TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
025416  013737  002260  002220                  MOV     L.CS1,E.CS1             ;SET EXPECTED CS1
025424  023737  002160  002220                  CMP     T.CS1,E.CS1             ;CHECK IF CS1 OK
025432  001345                                  BNE     2$                      ;NO - SKIP

025434  005724                                  TST     (R4)+                   ;BUMP RETURN POINTER

025436  005037  002326                          CLR     BITCNT                  ;CLEAR BIT COUNTER
025442  005037  002322                          CLR     M1.BIT                  ;CLEAR ALL WRITE BIT INDICATORS
025446  005037  002324                          CLR     M2.BIT
025452  005037  002316                          CLR     P1.BIT
```

```
025456  005037  002320                  CLR     PR.BIT
025462  012737  062040  002244          MOV     #DMD!ECCW!MEWD!WRTGATE,E.MR1 ;SET EXPECTED MR1
025470  012700  000400                  MOV     #256.,R0        ;SET COUNT TO WRITE 256 "0"
025474  004737  030652          5$:     JSR     PC,WRTBIT       ;CALL WRITE BIT
025500  000405                          BR      6$              ;ERROR RETURN - SKIP TO HANDLER
025502  005237  002326                  INC     BITCNT          ;BUMP BIT COUNT
025506  005300                          DEC     R0              ;LOOP UNTIL ALL 256
025510  001371                          BNE     5$              ;BITS ARE WRITTEN
025512  000402                          BR      7$
025514  010446                  6$:     MOV     R4,-(SP)        ;WRITE ERROR HANDLER-SET
025516  000204                          RTS     R4              ;STACK FOR REPORTING SUBSEQUENT ERRORS
025520  012737  000001  002316  7$:     MOV     #1,P1.BIT       ;SET SYNC BIT OF 1 AS NEXT BIT
025526  004737  030652                  JSR     PC,WRTBIT       ;GO WRITE IT
025532  000770                          BR      6$              ;ERROR RETURN
025534  005237  002326                  INC     BITCNT          ;BUMP BIT COUNT
025540  004437  032356                  JSR     R4,GETREG       ;GET RK611 REGS
025544  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
025546  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
025554  001274                          BNE     2$              ;NO - SKIP
025556  005724                          TST     (R4)+           ;BUMP PAST ERROR RETURNS
025560  000711                          BR      3$              ;GO TO EXIT
025562  032777  001000  153350  50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
025570  001403                          BEQ     51$             ;YES - SKIP
025572  005037  001236                  CLR     $ESCAPE         ;ELSE SET TO LOOP
025576  000702                          BR      3$
025600  010446                  51$:    MOV     R4,-(SP)        ;PRESET STACK FOR RETURN
025602  000204                          RTS     R4
                                .SBTTL  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
025604                          DWRITE:
025604  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025606  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025610  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025612  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
025614  005037  002326                  CLR     BITCNT          ;CLEAR BIT COUNTER
025620  005037  002342                  CLR     ECCLO           ;CLEAR ECC LOW
```

```
025624  005037  002340                  CLR     ECCHI                   ;CLEAR ECC HI
025630  012700  000400                  MOV     #400,R0                 ;SET WORD COUNT FOR ONE SECTOR
025634  012703  047636                  MOV     #OBUFF,R3               ;GET POINTER TO OUTPUT DATA
025640  010437  001216                  MOV     R4,$TMP6                ;STORE MR1 ERROR RETURN
025644  005724                          TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
025646  010437  001220                  MOV     R4,$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.

025652  005037  001214                  CLR     $TMP5                   ;CLEAR SWITCH
025656  012701  000001          3$:     MOV     #BIT0,R1                ;LOAD BIT POINTER
025662  012305                          MOV     (R3)+,R5                ;GET A WORD FOR WRITING

025664  013737  002322  002324  4$:     MOV     M1.BIT,M2.BIT           ;SHIFT CURRENT-1 TO CURRENT-2
025672  013737  002320  002322          MOV     PR.BIT,M1.BIT           ;PRESENT TO CURRENT-1
025700  013737  002316  002320          MOV     P1.BIT,PR.BIT           ;NEXT TO PRESENT
025706  005037  002316                  CLR     P1.BIT                  ;CLEAR NEXT BIT
025712  030105                          BIT     R1,R5                   ;TEST IF NEXT BIT IS ONE
025714  001403                          BEQ     5$                      ;NO - SKIP
025716  012737  000001  002316          MOV     #1,P1.BIT               ;ELSE SET NEXT BIT
025724  004737  030652          5$:     JSR     PC,WRTBIT               ;GO WRITE BIT
025730  000441                          BR      33$                     ;ERROR RETURN - SKIP
025732  004737  030462          43$:    JSR     PC,ECCGEN               ;GO GENERATE ECC FOR THIS BIT
025736  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;GET PATTERN
025744  023737  002254  002214          CMP     E.ECPT,T.ECPT           ;CHECK IF ECC CORRECT
025752  001402                          BEQ     40$                     ;YES - SKIP
025754  000137  026352                  JMP     13$                     ;JUMP TO ERROR EXIT

025760  005237  002326          40$:    INC     BITCNT                  ;BUMP BIT COUNT
025764  005701                          TST     R1                      ;TEST BIT POINTER
025766  100424                          BMI     7$                      ;IF NEGATIVE - SKIP
025770  006301                          ASL     R1                      ;ELSE SHIFT LEFT AND LOOP
025772  020027  000001                  CMP     R0,#1                   ;TEST IF WRITING LAST WORD
025776  001332                          BNE     4$                      ;NO - GO TO LOOP
026000  032737  010000  002260          BIT     #CFMT,L.CS1             ;TEST IF 22 SECTOR MODE
026006  001404                          BEQ     32$                     ;NO - SKIP
026010  005737  001214                  TST     $TMP5                   ;TEST IF 18 BIT SWITCH SET
026014  001001                          BNE     32$                     ;YES - SKIP
026016  000722                          BR      4$                      ;ELSE LOOP
026020  005701                  32$:    TST     R1                      ;ELSE TEST IF LAST BIT OF LAST WORD
026022  100320                          BPL     4$                      ;NO - GO TO LOOP
026024  042737  020000  002244          BIC     #ECCW,E.MR1             ;ELSE SET MR1 FOR ECC WRITE
026032  000714                          BR      4$
026034  000137  026374          33$:    JMP     20$

026040  032737  010000  002260  7$:     BIT     #CFMT,L.CS1             ;TEST IF 22 SECTOR MODE
026046  001413                          BEQ     31$                     ;NO - SKIP
026050  005737  001214                  TST     $TMP5                   ;TEST IF 18 BIT SWITCH SET
026054  001006                          BNE     30$                     ;YES - SKIP
026056  012701  040000                  MOV     #BIT14,R1               ;SET TO SUPPLY 2 MORE BITS (16 & 17)
026062  005005                          CLR     R5                      ;MAKE SURE THEY ARE 0'S
026064  005137  001214                  COM     $TMP5                   ;SET THE 18 BIT SWITCH
026070  000675                          BR      4$                      ;GO WRITE THE BITS
```

```
026072  005037  001214          30$:    CLR     $TMP5           ;CLEAR THE SWITCH
026076  005300                  31$:    DEC     R0              ;DEC WORD COUNT
026100  001266                          BNE     3$              ;IF NOT ZERO - GO GET NEXT WORD
026102  005037  001214                  CLR     $TMP5           ;CLEAR FOR PASS CONTROL
026106  013705  002340                  MOV     ECCHI,R5        ;GET FIRST ECC WORD
026112  012701  000001                  MOV     #BIT0,R1        ;SET BIT POINTER

026116  013737  002322  002324  8$:     MOV     M1.BIT,M2.BIT   ;SHIFT BIT INDICATORS AGAIN
026124  013737  002320  002322          MOV     PR.BIT,M1.BIT
026132  013737  002316  002320          MOV     P1.BIT,PR.BIT
026140  005037  002316                  CLR     P1.BIT          ;CLEAR NEXT BIT
026144  030105                          BIT     R1,R5           ;TEST IF BIT IS ZERO
026146  001403                          BEQ     9$              ;YES - SKIP
026150  012737  000001  002316          MOV     #1,P1.BIT       ;ELSE CHANGE TO ONE
026156  004737  030652          9$:     JSR     PC,WRTBIT       ;GO WRITE BIT
026162  000501                          BR      14$             ;WRTBIT ERROR - SKIP

026164  005237  002326          41$:    INC     BITCNT          ;BUMP BIT COUNT
026170  005701                          TST     R1              ;TEST BIT POINTER
026172  100413                          BMI     10$             ;IF NEGATIVE - SKIP
026174  006301                          ASL     R1              ;ELSE SHIFT AND LOOP
026176  023727  001214  000001          CMP     $TMP5,#1        ;TEST IF WRITING LAST ECC WORD
026204  001344                          BNE     8$              ;NO - SKIP TO LOOP
026206  005701                          TST     R1              ;ELSE TEST IF WRITING LAST BIT OF LAST ECC
026210  100342                          BPL     8$              ;NO - SKIP TO LOOP
026212  052737  020000  002244          BIS     #ECCW,E.MR1     ;ELSE SET ECCW IN MR1
026220  000736                          BR      8$

026222  005737  001214          10$:    TST     $TMP5           ;TEST PASS  0 JUST DONE
026226  001007                          BNE     11$             ;NO - SKIP
026230  005237  001214                  INC     $TMP5           ;BUMP PASS COUNT
026234  012701  000001                  MOV     #BIT0,R1        ;PRESET BIT POINTER
026240  013705  002342                  MOV     ECCLO,R5        ;GET SECOND ECC WORD
026244  000724                          BR      8$              ;LOOP

026246  012701  000017          11$:    MOV     #15.,R1         ;SET A BIT COUNT

026252  013737  002322  002324  12$:    MOV     M1.BIT,M2.BIT   ;SHIFT BIT INDICATORS
026260  013737  002320  002322          MOV     PR.BIT,M1.BIT
026266  013737  002316  002320          MOV     P1.BIT,PR.BIT
026274  005037  002316                  CLR     P1.BIT          ;SET ZERO FOR WRITE POSTAMBLE
026300  004737  030652                  JSR     PC,WRTBIT       ;GO WRITE BIT
026304  000436                          BR      15$             ;WRITE BIT ERROR - SKIP
026306  005237  002326          42$:    INC     BITCNT          ;BUMP BIT COUNT
026312  005301                          DEC     R1              ;DEC BIT COUNT
026314  001356                          BNE     12$             ;LOOP UNTTIL BIT COUNT 0

026316  004437  032356                  JSR     R4,GETREG       ;GET RK611 REGS
026322  013704  001220                  MOV     $TMP7,R4        ;SET R4 FOR ECC ERROR RETURN
026326  005724                          TST     (R4)+           ;BUMP TO CS1 ERROR RETURN
026330  013737  002260  002220          MOV     L.CS1,E.CS1     ;GET EXPECTED CS1

026336  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
026344  001031                          BNE     18$             ;NO - SKIP
026346  005724                          TST     (R4)+           ;BUMP TO GOOD RETURN
026350  000446                          BR      19$             ;GO TO RETURN
```

```

026352  013704  001220           13$:    MOV     $TMP7,R4                ;SET FOR ECC ERROR RETURN
026356  012737  025760  001236           MOV     #40$,$ESCAPE            ;STORE RETURN POINT FOR NO LOOP ON ERROR
026364  000410                           BR      16$

026366  013704  001216           14$:    MOV     $TMP6,R4                ;SET FOR MR1 ERROR RETURN
026372  000405                           BR      16$

026374  013704  001216           20$:    MOV     $TMP6,R4                ;SET FOR MR1 ERROR RETURN
026400  000402                           BR      16$

026402  013704  001216           15$:    MOV     $TMP6,R4                ;SET FOR MR1 ERROR RETURN

026406  032777  001000  152524   16$:    BIT     #BIT9,@SWR              ;CHECK IF LOOP ON ERROR
026414  001403                           BEQ     17$                     ;NO - SKIP

026416  005037  001236                   CLR     $ESCAPE                 ;CLEAR ESCAPE
026422  000421                           BR      19$                     ;SKIP TO EXIT

026424  010446                   17$:    MOV     R4,-(SP)                ;CONDITION STACK
026426  000204                           RTS     R4                      ;GO REPORT ERROR

026430  005037  001236           18$:    CLR     $ESCAPE                 ;CLEAR FOR LOOP OR EXIT
026434  032777  001000  152476           BIT     #BIT9,@SWR              ;CHECK IF LOOP ON ERROR
026442  001011                           BNE     19$                     ;YES - SKIP
026444  013700  001254                   MOV     $TESTN,R0               ;SET UP TO SKIP TO NEXT TEST
026450  006300                           ASL     R0
026452  016037  033472  001236           MOV     $SW08TB(R0),$ESCAPE
026460  162737  000002  001236           SUB     #2,$ESCAPE
026466                           19$:
026466  012605                           MOV     (SP)+,R5                ;;POP STACK INTO R5
026470  012603                           MOV     (SP)+,R3                ;;POP STACK INTO R3
026472  012601                           MOV     (SP)+,R1                ;;POP STACK INTO R1
026474  012600                           MOV     (SP)+,R0                ;;POP STACK INTO R0
026476  000204                           RTS     R4

                                 .SBTTL  DIAGNOSTIC MODE READ GAP AND DATA SYNC
                                 ;*      THE READING OF THE GAP AND READING OF THE SYNC (OR PREAMBLE)
                                 ;*      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

026500                           DRGPSN:
026500  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.
 026502  005737  002266                         TST     L.DA                    ;TEST IF DESIRED TRACK/SECTOR NOW 0
 026506  001006                                 BNE     8$                      ;NO - SKIP
 026510  012700  000066                         MOV     #54.,R0                 ;ELSE ADJUST COUNT FOR CYL CHANGE
 026514  012737  000011  002326                 MOV     #11,BITCNT              ;PRESET BIT COUNT FOR REST OF GAP
 026522  000416                                 BR      10$
 026524  105737  002266                 8$:     TSTB    L.DS                    ;TEST IF NEW SECTOR IS 0
 026530  001006                                 BNE     9$                      ;NO - SKIP
 026532  012700  000070                         MOV     #56.,R0                 ;ELSE ADJUST COUNT FOR TRACK CHANGE
 026536  012737  000007  002326                 MOV     #7,BITCNT               ;PRESET BIT COUNT FOR REST OF GAP
 026544  000405                                 BR      10$
 026546  012737  000005  002326         9$:     MOV     #5,BITCNT               ;PRESET BIT COUNTER FOR REST OF GAP
 026554  012700  000074                         MOV     #60.,R0                 ;SET COUNT FOR 60 BITS
                                        ;       END OF BLOCK

 026560  012737  022040  002244         10$:    MOV     #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
 026566  005037  002320                         CLR     PR.BIT                  ;CLEAR PRESENT BIT
 026572  005037  002322                         CLR     M1.BIT                  ;CLEAR PREVIOUS BIT
 026576  004737  031616                 1$:     JSR     PC,RDBIT                ;GO READ BIT
 026602  016237  000026  002204                 MOV     RKMR1(R2),T.MR1         ;GET MR1
 026610  023737  002244  002204                 CMP     E.MR1,T.MR1             ;CHECK IF CORRECT
 026616  001404                                 BEQ     11$                     ;YES - SKIP
 026620  012737  026642  001236                 MOV     #2$,$ESCAPE             ;SET RETURN TO CONT TEST
 026626  000552                                 BR      50$
 026630  005237  002326                 11$:    INC     BITCNT                  ;BUMP BIT COUNT
 026634  005300                                 DEC     R0                      ;DEC LOOP COUNT
 026636  001357                                 BNE     1$                      ;LOOP IF NOT ZERO
 026640  000421                                 BR      4$                      ;ELSE SKIP

 026642  005037  001236                 2$:     CLR     $ESCAPE                 ;CLEAR TO EXIT OR LOOP
 026646  022777  001000  152264                 CMP     #BIT9,@SWR              ;ERROR RETURN: CHECK IF LOOP ON ERROR
 026654  001011                                 BNE     3$                      ;YES - SKIP
 026656  013700  001254                         MOV     $TESTN,R0               ;GET NUMBER OF NEXT TEST
 026662  006300                                 ASL     R0                      ;SHIFT FOR INDEXING
 026664  016037  033472  001236                 MOV     $SW08TB(R0),$ESCAPE  ;SET ESCAPE FOR GOT TO NEXT TEST
 026672  162737  000002  001236                 SUB     #2,$ESCAPE              ;SET TO FIRST INST (SCOPE)
 026700                                 3$:
 026700  012600                                 MOV     (SP)+,R0                ;;POP STACK INTO R0
 026702  000204                                 RTS     R4

 026704  004437  032356                 4$:     JSR     R4,GETREG               ;GET RK611 REGS
 026710  005724                                 TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
 026712  013737  002260  002220                 MOV     L.CS1,E.CS1             ;SET EXPECTED CS1
 026720  023737  002160  002220                 CMP     T.CS1,E.CS1             ;CHECK IF CS1 OK
 026726  001345                                 BNE     2$                      ;NO - SKIP

 026730  005724                                 TST     (R4)+                   ;BUMP RETURN POINTER

 026732  005037  002326                         CLR     BITCNT                  ;CLEAR BIT COUNTER
 026736  012737  022040  002244                 MOV     #DMD!ECCW!MEWD,E.MR1 ;SET EXPECTED MR1
 026744  012700  000200                         MOV     #128.,R0                ;SET COUNT TO READ 128 "0"

 026750  004737  031616                 5$:     JSR     PC,RDBIT                ;CALL READ BIT
 026754  016237  000026  002204                 MOV     RKMR1(R2),T.MR1 ;GET MR1 FOR TESTING
```

```
026762  023737  002244  002204          CMP     E.MR1,T.MR1     ;TEST IF CORRECT
026770  001404                          BEQ     22$             ;YES - SKIP

026772  012737  027002  001236          MOV     #22$,$ESCAPE    ;SET RETURN FOR CONTINUE TEST
027000  000465                          BR      50$             ;GO TO ERROR EXIT
027002  005237  002326          22$:    INC     BITCNT          ;BUMP BIT COUNT
027006  005300                          DEC     R0              ;LOOP UNTIL ALL 128
027010  001357                          BNE     5$              ;BITS ARE READ
027012  012700  000177                  MOV     #127.,R0        ;SET COUNT FOR REST OF SYNC
027016  052737  100000  002244          BIS     #RDGATE,E.MR1   ;SET EXPECTED MR1
027024  004737  031616          6$:     JSR     PC,RDBIT        ;GO CALL READ BIT
027030  016237  000026  002204          MOV     RKMR1(R2),T.MR1 ;GET MR1 FOR TESTING
027036  023737  002244  002204          CMP     E.MR1,T.MR1     ;TEST IF CORRECT
027044  001404                          BEQ     23$             ;YES - SKIP
027046  012737  027056  001236          MOV     #23$,$ESCAPE    ;ELSE SET ESCAPE
027054  000437                          BR      50$             ;GO TO ERROR REPORT EXIT

027056  005237  002326          23$:    INC     BITCNT          ;BUMP BIT COUNTER
027062  005300                          DEC     R0              ;DEC BIT COUNT
027064  001357                          BNE     6$              ;LOOP UNTIL ALL 127 READ

027066  012737  000001  002320          MOV     #1,PR.BIT       ;SET SYNC BIT OF 1 AS NEXT BIT
027074  004737  031616                  JSR     PC,RDBIT        ;GO READ IT
027100  016237  000026  002244          MOV     RKMR1(R2),E.MR1 ;GET MR1 FOR TESTING
027106  023737  002244  002204          CMP     E.MR1,T.MR1     ;CHECK IF CORRECT
027114  001404                          BEQ     24$             ;YES - SKIP
027116  012737  027126  001236          MOV     #24$,$ESCAPE    ;ELSE SET RETURN FOR CONTINUE TEST
027124  000413                          BR      50$

027126  005237  002326          24$:    INC     BITCNT          ;BUMP BIT COUNT

027132  004437  032356                  JSR     R4,GETREG       ;GET RK611 REGS
027136  005724                          TST     (R4)+           ;BUMP TO NEXT ERROR RETURN
027140  023737  002160  002220          CMP     T.CS1,E.CS1     ;CHECK IF CORRECT
027146  001235                          BNE     2$              ;NO - SKIP

027150  005724                          TST     (R4)+           ;BUMP PAST ERROR RETURNS
027152  000652                          BR      3$              ;GO TO EXIT

027154  032777  001000  151756  50$:    BIT     #BIT9,@SWR      ;TEST IF LOOP ON ERROR
027162  001403                          BEQ     51$             ;YES - SKIP
027164  005037  001236                  CLR     $ESCAPE         ;ELSE SET TO LOOP
027170  000643                          BR      3$
027172  010446                  51$:    MOV     R4,-(SP)        ;PRESET STACK FOR RETURN
027174  000204                          RTS     R4
                                .SBTTL  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 SECTOR.
                                ;*      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
                                       ;*

      027176                           DWRTCK:
      027176                           DREAD:
      027176  010046                           MOV     R0,-(SP)                ;;PUSH R0 ON STACK
      027200  010146                           MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      027202  010346                           MOV     R3,-(SP)                ;;PUSH R3 ON STACK
      027204  010546                           MOV     R5,-(SP)                ;;PUSH R5 ON STACK
      027206  012400                           MOV     (R4)+,R0                ;STORE WORD COUNT
      027210  010437  001216                   MOV     R4,$TMP6                ;STORE MR1 ERROR RETURN
      027214  005724                           TST     (R4)+                   ;BUMP TO NEXT RETURN
      027216  010437  001220                   MOV     R4,$TMP7                ;STORE ECC ERROR RETURN
      027222  005724                           TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
      027224  010437  001210                   MOV     R4,$TMP3                ;STORE CS1 ERROR RETURN
      027230  005724                           TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
      027232  010437  001212                   MOV     R4,$TMP4                ;STORE ECC REG ERROR RETURN
      027236  005724                           TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
```

```
027240  010437  001222                  MOV     R4,$TMP10               ;STORE ECC PAT ERR RETURN
027244  005724                          TST     (R4)+                   ;BUMP TO NEXT ERROR RETURN
027246  010437  001224                  MOV     R4,$TMP11               ;SOTER ECC POS ERROR RETURN
027252  012703  047636                  MOV     #OBUFF,R3               ;GET ADDRESS OF BUFFER
027256  005037  002326                  CLR     BITCNT                  ;CLEAR BIT COUNTER
027262  005037  002340                  CLR     ECCHI                   ;CLEAR ECC COMPUTED LOCATIONS
027266  005037  002342                  CLR     ECCLO
027272  005037  001214                  CLR     $TMP5                   ;CLEAR FOR 18 BIT MODE SWITCH
027276  032737  010000  002260          BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
027304  001412                          BEQ     32$                     ;NO - SKIP
027306  012737  011000  001226          MOV     #11000,$TMP12           ;ELSE SET UP BIT COUNTS FOR 18 BIT MODE
027314  012737  011001  001230          MOV     #11001,$TMP13
027322  012737  011041  001232          MOV     #11041,$TMP14
027330  000411                          BR      3$
027332  012737  010000  001226  32$:    MOV     #10000,$TMP12           ;SET BIT COUNTS FOR 16 BIT MODE
027340  012737  010001  001230          MOV     #10001,$TMP13
027346  012737  010041  001232          MOV     #10041,$TMP14

027354  012701  000001          3$:     MOV     #BIT0,R1                ;PRESET BIT POINTER
027360  012305                          MOV     (R3)+,R5                ;GET DATA WORD

027362  013737  002320  002322  4$:     MOV     PR.BIT,M1.BIT           ;SHIFT PRESENT INTO PREVIOUS
027370  005037  002320                  CLR     PR.BIT                  ;CLEAR PRESENT
027374  030105                          BIT     R1,R5                   ;TEST IF PRESENT BIT IS 0
027376  001403                          BEQ     1$                      ;YES - SKIP
027400  012737  000001  002320          MOV     #1,PR.BIT               ;ELSE INSERT A 1

027406  004737  031616          1$:     JSR     PC,RDBIT                ;GO READ BIT
027412  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;GET MR1
027420  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF CORRECT
027426  001402                          BEQ     43$                     ;YES - SKIP
027430  000137  030262                  JMP     9$

027434  005737  002326          43$:    TST     BITCNT                  ;TEST IF FIRST BIT OF DATA
027440  001413                          BEQ     44$                     ;YES - SKIP ECC GEN
027442  004737  030462                  JSR     PC,ECCGEN               ;GO GENERATE ECC
027446  016237  000032  002214          MOV     RKECPT(R2),T.ECPT       ;TEST IF ECC CORRECT
027454  023737  002214  002254          CMP     T.ECPT,E.ECPT
027462  001402                          BEQ     44$
027464  000137  030276                  JMP     10$

027470  005237  002326          44$:    INC     BITCNT                  ;BUMP BIT COUNTER
027474  005701                          TST     R1                      ;TEST IF THIS WORD IS DONE
027476  100402                          BMI     2$                      ;YES - SKIP
027500  006301                          ASL     R1                      ;ELSE SHIFT BIT POINTER
027502  000727                          BR      4$

027504  032737  010000  002260  2$:     BIT     #CFMT,L.CS1             ;TEST IF 18 BIT MODE
027512  001412                          BEQ     30$                     ;NO - SKIP
027514  005737  001214                  TST     $TMP5                   ;TEST IF PROCESSING BIT 16 OR 17 DONE
027520  001007                          BNE     30$                     ;YES - SKIP
027522  012701  040000                  MOV     #BIT14,R1               ;SET POINTER FOR 2 MORE BITS
027526  013705  002354                  MOV     HIBITS,R5               ;INSERT HI ORDER BITS
027532  005137  001214                  COM     $TMP5                   ;SET 18 BIT SWITCH
027536  000711                          BR      4$
027540  005037  001214          30$:    CLR     $TMP5                   ;CLEAR 18 BIT SWITCH
```

```
 027544  005300                          31$:    DEC     R0                      ;DECREMENT WORD COUNT
 027546  001302                                  BNE     3$                      ;IF NOT ZERO - LOOP
 027550  023737  002326  001226                  CMP     BITCNT,$TMP12           ;ELSE TEST IF FULL SECTOR WAS READ
 027556  001402                                  BEQ     33$                     ;YES - SKIP
 027560  000137  030234                          JMP     8$                      ;ELSE JUMP TO EXIT - DO NOT CHECK
                                                                                 ;       ECC WORDS
 027564  013737  002320  002322          33$:    MOV     PR.BIT,M1.BIT           ;MOVE LAST DATA BIT FOR ECC GEN
 027572  004737  030462                          JSR     PC,ECCGEN               ;GENERATE ECC
 027576  005037  001214                          CLR     $TMP5                   ;CLEAR FOR PASS COUNTING
 027602  005737  002344                          TST     ECCSRC                  ;TEST WHERE ECC WORDS ARE
 027606  001006                                  BNE     36$                     ;THEY ARE IN BUFFER - SKIP
 027610  013737  002340  050636                  MOV     ECCHI,OBUFF+1000                ;ELSE MOVE THEM INTO BUFFER
 027616  013737  002342  050640                  MOV     ECCLO,OBUFF+1002
 027624  012305                          36$:    MOV     (R3)+,R5                ;GET ECC FROM BUFFER
 027626  012701  000001                          MOV     #BIT0,R1                ;SET BIT POINTER
 027632  013737  002320  002322          5$:     MOV     PR.BIT,M1.BIT           ;SHIFT PRESENT TO PREVIOUS
 027640  005037  002320                          CLR     PR.BIT                  ;AND SET UP PRESENT
 027644  030105                                  BIT     R1,R5
 027646  001403                                  BEQ     34$
 027650  012737  000001  002320                  MOV     #1,PR.BIT
 027656  004737  031616                  34$:    JSR     PC,RDBIT                ;GO READ BIT
 027662  016237  000026  002204                  MOV     RKMR1(R2),T.MR1         ;GET MR1
 027670  023737  002204  002244                  CMP     T.MR1,E.MR1             ;CHECK IF CORRECT
 027676  001402                                  BEQ     38$                     ;YES - SKIP
 027700  000137  030312                          JMP     11$                     ;ELSE JUMP TO REPORT EXIT
 027704  023737  002326  001226          38$:    CMP     BITCNT,$TMP12           ;TEST IF FIRST ECC BIT
 027712  001402                                  BEQ     49$                     ;YES - SKIP ECC GEN (ALREADY DONE)
 027714  004737  030462                          JSR     PC,ECCGEN               ;GENERATE ECC FOR BIT
 027720  016237  000032  002214          49$:    MOV     RKECPT(R2),T.ECPT               ;GET PATTERN
 027726  023737  002214  002254                  CMP     T.ECPT,E.ECPT           ;TEST IF ECC PAT CORRECT
 027734  001402                                  BEQ     45$                     ;YES SKIP
 027736  000137  030276                          JMP     10$                     ;ELSE GO REPORT ERROR
 027742  005237  002326                  45$:    INC     BITCNT                  ;BUMP BIT COUNT
 027746  023737  002326  001230                  CMP     BITCNT,$TMP13           ;TEST IF 1ST BIT OF ECC
 027754  001003                                  BNE     46$                     ;NO - SKIP
 027756  042737  020000  002244                  BIC     #ECCW,E.MR1             ;ELSE CLR ECCW FOR ECC READ
 027764  023737  002326  001232          46$:    CMP     BITCNT,$TMP14           ;TEST IF 1ST BIT OF POSTAMBLE
 027772  001027                                  BNE     47$                     ;NO - SKIP
 027774  052737  020000  002244                  BIS     #ECCW,E.MR1             ;ELSE SET ECCW FOR ECC DONE
 030002  042737  100000  002244                  BIC     #RDGATE,E.MR1           ;CLEAR READ GATE
 030010  016237  000032  002214                  MOV     RKECPT(R2),T.ECPT ;STORE ACTUAL PATTERN
 030016  016237  000030  002212                  MOV     RKECPS(R2),T.ECPS ;STORE ACTUAL POSITION
 030024  013737  002350  002252                  MOV     ECPOSX,E.ECPS           ;GET EXPECTED POSITION FOR REPORT
 030032  023737  002214  002254                  CMP     T.ECPT,E.ECPT           ;CHECK IF CORRECT
 030040  001163                                  BNE     20$                     ;NO SKIP
 030042  023737  002212  002252                  CMP     T.ECPS,E.ECPS           ;CHECK IF POSITION CORRECT
 030050  001157                                  BNE     20$                     ;NO - SKIP
 030052  023737  002326  001232          47$:    CMP     BITCNT,$TMP14           ;TEST IF ECC DONE
```

```
030060  002436                          BLT     48$                     ;NO - SKIP
030062  005737  002346                  TST     ECPATX                  ;TEST IF EXPECTED ECC IS 0
                                                                        ;(NO ERROR)
030066  001433                          BEQ     48$                     ;YES - SKIP
030070  023737  001232  002326          CMP     $TMP14,BITCNT           ;TEST IF FIRST POSTAMBLE BIT
030076  001402                          BEQ     52$                     ;YES - SKIP
030100  005237  002350                  INC     ECPOSX                  ;ELSE BUMP POS COUNT EXPECTED
030104  016237  000032  002214  52$:    MOV     RKECPT(R2),T.ECPT               ;GET PATTERN
030112  016237  000030  002212          MOV     RKECPS(R2),T.ECPS               ;GET POSITION
030120  013737  002350  002252          MOV     ECPOSX,E.ECPS           ;GET EXPECTED POSITION
030126  023737  002214  002254          CMP     T.ECPT,E.ECPT           ;TEST IF PATTERN CORRECT
030134  001402                          BEQ     51$                     ;YES - SKIP
030136  000137  030326                  JMP     22$                     ;ELSE GO REPORT
030142  023737  002212  002252  51$:    CMP     T.ECPS,E.ECPS           ;CHECK IF POSITION CORRECT
030150  001402                          BEQ     48$                     ;YES - SKIP
030152  000137  030342                  JMP     23$                     ;ELSE GO REPORT

030156  005701                  48$:    TST     R1                      ;TEST IF WORD DONE
030160  100403                          BMI     6$                      ;YES SKIP
030162  006301                          ASL     R1                      ;ELSE SHIFT BIT POINTER
030164  000137  027632                  JMP     5$                      ;AND LOOP

030170  005737  001214          6$:     TST     $TMP5                   ;TEST PASS COUNT. IF NOT 0
                                                                        ;(NOT THE FIRST ECC WORD JUST WRITTE)
030174  001004                          BNE     7$                      ; - SKIP

030176  005237  001214                  INC     $TMP5                   ;BUMP PASS COUNT
030202  000137  027624                  JMP     36$

030206  023727  001214  000001  7$:     CMP     $TMP5,#1                ;TEST PASS COUNT. IF NOT 1 (NOT THE
030214  001007                          BNE     8$                      ;2ND ECC WORD JUST READ) - SKIP
030216  005005                          CLR     R5                      ;CLEAR WORD (POSTAMBLE)
030220  012701  000001                  MOV     #BIT0,R1                ;SET BIT POINTER
030224  005237  001214                  INC     $TMP5                   ;BUMP PASS COUNT
030230  000137  027632                  JMP     5$

030234  004437  032356          8$:     JSR     R4,GETREG               ;GET RK611 REGS
030240  013737  002260  002220          MOV     L.CS1,E.CS1             ;SET EXPECTED CS1
030246  023737  002220  002160          CMP     E.CS1,T.CS1             ;CHECK IF CORRECT
030254  001060                          BNE     21$                     ;NO - SKIP
030256  005724                          TST     (R4)+                   ;ELSE BUMP RETURN POINTER TO NO ERR
030260  000446                          BR      13$

030262  013704  001216          9$:     MOV     $TMP6,R4                ;SET RETURN TOR MR1 ERR.
030266  012737  027434  001236          MOV     #43$,$ESCAPE            ;SET RETURN TO CONTINUE TEST
030274  000427                          BR      12$

030276  013704  001220          10$:    MOV     $TMP7,R4                ;SET RETURN FOR ECC ERROR
030302  012737  027470  001236          MOV     #44$,$ESCAPE            ;SET RETURN TO CONTINUE TEST
030310  000421                          BR      12$

030312  013704  001216          11$:    MOV     $TMP6,R4                ;SET RETURN FOR  MR1 ERROR
030316  012737  027742  001236          MOV     #45$,$ESCAPE            ;SET RETURN TO CONTINUE TEST
030324  000413                          BR      12$

030326  013704  001222          22$:    MOV     $TMP10,R4               ;GET RETURN FOR PATTERN ERROR
```

```
030332  012737  030142  001236           MOV     #51$,$ESCAPE       ;SET RETURN TO CONTINUE TEST
030340  000405                           BR      12$
030342  013704  001224          23$:     MOV     $TMP11,R4          ;RET RETURN FOR POSITION ERROR
030346  012737  030156  001236           MOV     #48$,$ESCAPE       ;SET RETURN TO CONTINUE TEST
030354  013746  001224          12$:     MOV     $TMP11,-(SP)       ;PREP STACK FOR RETURN
030360  032777  001000  150552           BIT     #BIT9,@SWR         ;TEST IF LOOP ON ERROR
030366  001407                           BEQ     14$                ;NO SKIP
030370  005726                           TST     (SP)+              ;REMOVE LAST ENTRY FROM STACK
030372  005037  001236                   CLR     $ESCAPE            ;NO LOOP - SLEAR ESCAPE
030376                          13$:
030376  012605                           MOV     (SP)+,R5           ;;POP STACK INTO R5
030400  012603                           MOV     (SP)+,R3           ;;POP STACK INTO R3
030402  012601                           MOV     (SP)+,R1           ;;POP STACK INTO R1
030404  012600                           MOV     (SP)+,R0           ;;POP STACK INTO R0
030406  000204                  14$:     RTS     R4
030410  013704  001212          20$:     MOV     $TMP4,R4           ;GET RETURN FOR ECC ERROR
030414  000402                           BR      15$
030416  013704  001210          21$:     MOV     $TMP3,R4           ;GET CS1 ERROR RETURN
030422  005037  001236          15$:     CLR     $ESCAPE            ;TEST IF LOOP ON ERROR
030426  032777  001000  150504           BIT     #BIT9,@SWR         ;TEST IF LOOP ON ERROR
030434  001360                           BNE     13$                ;YES - SKIP
030436  013700  001254                   MOV     $TESTN,R0          ;SET REGISTERS TO LOOP ON ERROR OR
030442  006300                           ASL     R0                 ;ESCAPE TO NEXT TEST.
030444  016037  033472  001236           MOV     $SWOBTB(R0),$ESCAPE      ;SET UP TO ESCAPE TO NEXT TEST
030452  162737  000002  001236           SUB     #2,$ESCAPE         ;ADJUST RETURN
030460  000746                           BR      13$                ;GO TO EXIT
                                .SBTTL   GENERATE ECC WORD
                                ;*       EACH TIME THIS ROUTINE IS CALLED THE ECC IS CALCULATED FOR THE
                                ;*       NEXT BIT TO BE READ OR WRITTEN AND THE BITS PREVIOUSLY
                                ;*       READ OR WRITTEN (STORED IN ECCHI AND ECCLO). THE RESULTS OF
                                ;*       THE CALCULATION ARE PLACED IN ECCHI AND ECCLO. THE LOW ORDER
                                ;*       11 BITS OF ECCHI ARE PLACED IN E.ECPT FOR EASY COMPARISON
                                ;*       TO RKECPT.
                                ;*
                                ;*       CALL:    JSR      PC,ECCGEN
                                ;*       RETURN:  RTS      R4
030462  010046                  ECCGEN:  MOV     R0,-(SP)           ;STORE R0
030464  012700  002316                   MOV     #P1.BIT,R0         ;GET ADDRESS OF NEXT BIT
030470  032737  000002  002260           BIT     #BIT1,L.CS1        ;CHECK IF WRITING
030476  001002                           BNE     12$                ;YES - SKIP
030500  012700  002322                   MOV     #M1.BIT,R0         ;ELSE CHANGE TO PRESENT BIT
030504  005710                  12$:     TST     (R0)               ;CHECK IF NEXT BIT 0
030506  001410                           BEQ     3$                 ;YES - SKIP
030510  032737  000001  002340           BIT     #1,ECCHI           ;NO, CHECK IF BIT 31 = 1
030516  001410                           BEQ     5$                 ;NO, SET ECC XORED BIT
030520  005037  002336          1$:      CLR     ECCXOR             ;CLEAR ECC XORED BIT
```

```
030524  000241                                  CLC                             ;CLEAR CARRY FLOP
030526  000410                                  BR      7$                      ;SHIFT IN ECC BIT

030530  032737  000001  002340  3$:             BIT     #1,ECCHI                ;CHECK IF BIT 31 = 1
030536  001770                                  BEQ     1$                      ;NO, CLEAR ECC XORED BIT
030540  012737  000001  002336  5$:             MOV     #1,ECCXOR               ;SET ECC XOR
030546  000261                                  SEC                             ;SET CARRY FLOP
030550  006037  002342          7$:             ROR     ECCLO
030554  006037  002340                          ROR     ECCHI
030560  005737  002336                          TST     ECCXOR                  ;CHECK IF XOR NEEDED
030564  001422                                  BEQ     10$                     ;NO, RETURN
030566  012746  020020                          MOV     #BIT13!BIT4,-(SP)       ;DO XOR OF ECC BITS
030572  043716  002342                          BIC     ECCLO,(SP)              ;  0, 2, 11, 21, 23
030576  042737  020020  002342                  BIC     #BIT13!BIT4,ECCLO
030604  052637  002342                          BIS     (SP)+,ECCLO
030610  012746  002400                          MOV     #BIT10!BIT8,-(SP)
030614  043716  002340                          BIC     ECCHI,(SP)
030620  042737  002400  002340                  BIC     #BIT10!BIT8,ECCHI
030626  052637  002340                          BIS     (SP)+,ECCHI
030632  013737  002340  002254  10$:            MOV     ECCHI,E.ECPT            ;STORE ECC PATTERN
030640  042737  174000  002254                  BIC     #174000,E.ECPT          ;MASK UNSEEN BITS
030646  012600                                  MOV     (SP)+,R0                ;RESTORE R0
030650  000207                                  RTS     PC                      ;RETURN

                                        .SBTTL  SIMULATE ONE BIT OF WRITE DATA IN MAINTANENCE MODE
                                        ;*      ONE BIT OF DATA IS WRITE SIMULATED. THE CONTENTS OF
                                        ;*      RKMR1 IS COMPUTED (PRECOMP ADVANCE AND DELAY, MAINTENANCE
                                        ;*      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 THE
                                        ;*      ERROR OCCURRED.
                                        ;*
                                        ;*      CALL:   JSR     PC,WRTBIT
                                        ;*      RETURN: RTS     PC+2            FOR NO ERROR RETURN
                                        ;*              RTS     PC              FOR ERROR IN MR1
030652  052737  002400  002244  WRTBIT: BIS     #MCLK!MEWD,E.MR1 ;CREATE EXPECTED MAINT. REG. 1
030660  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2) ;PROVIED 1ST UPWARD TRANSITION
030666  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MAINT. REG. 1
030674  023737  002244  002204          CMP     E.MR1,T.MR1             ;CHECK IF MAINT REG 1 CORRECT
030702  001416                          BEQ     3$                      ;YES, PROVIDE DOWNWARD TRANSISTION
030704  012737  030724  001236          MOV     #1$,$ESCAPE             ;LOAD ESCAPE FOR LOOP ON ERROR
030712  012737  043644  001512          MOV     #EMW1,EMW+2             ;LOAD ERROR MESSAGE
030720  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
030722  000207                          RTS     PC                      ;MR1 INCORRECT ON UPWARD TRANSITION

030724  032777  001000  150206  1$:     BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
030732  001402                          BEQ     3$                      ;NO, CONTINUE
030734  000137  031606                  JMP     63$                     ;YES, LOOP ON ERROR

030740  042737  014400  002244  3$:     BIC     #MCLK!PCA!PCD,E.MR1 ;INITIALIZE MAINTENANCE REG. 1
030746  052737  042000  002244          BIS     #MEWD!WRTGAT,E.MR1
030754  005737  002320                  TST     PR.BIT                  ;CHECK IF ONE
030760  001152                          BNE     20$                     ;YES, SIMULATE ONE
030762  005737  002322                  TST     M1.BIT                  ;CHECK IF PREVIOUS ONE
030766  001023                          BNE     10$                     ;YES, NO TRANSISTION
```

```
030770  042737  002000  002244          BIC     #MEWD,E.MR1             ;INDICATE TRANSISTION
030776  005737  002316                  TST     P1.BIT                  ;CHECK IF NEXT BIT = 1
031002  001007                          BNE     5$                      ;YES, CHECK FOR PRECOMP ADVANCE
031004  005737  002324                  TST     M2.BIT                  ;CHECK FOR PRECOMP. ADVANCE
031010  001412                          BEQ     10$                     ;NO, CLOCK IN ZERO
031012  052737  010000  002244          BIS     #PCD,E.MR1              ;SET PRECOMP. DELATY
031020  000406                          BR      10$                     ;CLOCK IN ZERO

031022  005737  002324          5$:     TST     M2.BIT                  ;CHECK FOR PRECOMP. ADVANCE
031026  001003                          BNE     10$                     ;CLOCK IN ZERO
031030  052737  004000  002244          BIS     #PCA,E.MR1              ;SET PRECOMP. ADVANCE
031036  012762  000040  000026  10$:    MOV     #DMD,RKMR1(R2)          ;CLOCK IN DATA BIT
031044  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MR1
031052  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF MR1 CORRECT
031060  001416                          BEQ     12$                     ;YES, CONTINUE
031062  012737  031102  001236          MOV     #11$,$ESCAPE            ;LOAD ESCAPE FOR LOOP ON ERROR
031070  012737  043732  001512          MOV     #EMW2,EMW+2             ;LOAD ERROR MESSAGE
031076  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031100  000207                          RTS     PC                      ;MR1 INCORRECT

031102  032777  001000  150030  11$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031110  001402                          BEQ     12$                     ;NO, CONTINUE
031112  000137  031606                  JMP     63$                     ;YES, LOOP ON ERROR

031116  052737  002400  002244  12$:    BIS     #MCLK!MEWD,E.MR1        ;CREATE EXPECTED MAINT REG 1
031124  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2)     ;PROVIDE 2ND UPWARD TRANSISITION
031132  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MAINT REG. 1
031140  023737  002244  002204          CMP     E.MR1,T.MR1             ;CHECK IF MAINT REG. 1 CORRECT
031146  001416                          BEQ     15$                     ;YES, CONTINUE
031150  012737  031170  001236          MOV     #13$,$ESCAPE            ;LOAD ESCAPE FOR LOOP ON ERROR
031156  012737  044022  001512          MOV     #EMW3,EMW+2             ;LOAD ERROR MESSAGE
031164  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031166  000207                          RTS     PC                      ;MR1 INCORRECT

031170  032777  001000  147742  13$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031176  001402                          BEQ     15$                     ;NO, CONTINUE
031200  000137  031606                  JMP     63$                     ;YES, LOOP ON ERROR

031204  052737  002000  002244  15$:    BIS     #MEWD,E.MR1             ;RESET TRANSITION INDICATION
031212  042737  000400  002244          BIC     #MCLK,E.MR1
031220  012762  000040  000026          MOV     #DMD,RKMR1(R2)          ;SUPPLY LAST PART OF DATA
031226  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MR1
031234  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF MR1 CORRECT
031242  001414                          BEQ     18$                     ;YES, RETURN
031244  012737  031264  001236          MOV     #17$,$ESCAPE            ;LOAD ESCAPE FOR LOOP ON ERROR
031252  012737  044110  001512          MOV     #EMW4,EMW+2             ;LOAD ERROR MESSAGE
031260  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031262  000207                          RTS     PC                      ;MR1 INCORRECT

031264  032777  001000  147646  17$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031272  001145                          BNE     63$                     ;YES, LOOP ON ERROR
031274  005037  001236          18$:    CLR     $ESCAPE                 ;CLEAR ESCAPE
031300  062716  000002                  ADD     #2,(SP)                 ;ADJUST RETURN
031304  000207                          RTS     PC                      ;RETURN

031306  005737  002316          20$:    TST     P1.BIT                  ;CHECK IN NEXT BIT A ONE
```

```
031312  001007                          BNE     30$                     ;YES, CHECK IF PRECOMP DELAY
031314  005737  002322                  TST     M1.BIT                  ;CHECK FOR PRECOMP ADVANCE
031320  001415                          BEQ     40$                     ;NO, CHECK FO MR1
031322  052737  004000  002244          BIS     #PCA,E.MR1              ;SET PRECOMP. ADVZNCE
031330  000411                          BR      40$                     ;CHECK MR1

031332  042737  000407  002244  30$:    BIC     #MCLK,E.MR1             ;RESET MAINT CLOCK IN EXPECTED MR1
031340  005737  002322                  TST     M1.BIT                  ;CHECK FOR PRECOMP DELAY
031344  001003                          BNE     40$                     ;NO, CHECK MR1
031346  052737  010000  002244          BIS     #PCD,E.MR1              ;SET SET PRECOMP DELAY
031354  012762  000040  000026  40$:    MOV     #DMD,RKMR1(R2)          ;CLOCK IN DATA BIT
031362  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MR1
031370  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK MR1 CORRECT
031376  001414                          BEQ     42$                     ;YES, CLOCK IN REST OF BIT
031400  012737  031420  001236          MOV     #41$,$ESCAPE            ;LOAD ESCAPE FOR LOOP ON ERROR
031406  012737  043732  001512          MOV     #EMW2,EMW+2             ;LOAD ERROR MESSAGE
031414  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031416  000207                          RTS     PC                      ;MR1 INCORRECT

031420  032777  001000  147512  41$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031426  001067                          BNE     63$                     ;YES, LOOP ON ERROR
031430  052737  000400  002244  42$:    BIS     #MCLK,E.MR1             ;CHREATE EXPECTED MAINT. REG. 1
031436  012762  000440  000026          MOV     #DMD!MCLK,RKMR1(R2)     ;PROVIDE 2ND UPWARD TRANSITION
031444  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MAINT REG 1
031452  023737  002244  002204          CMP     E.MR1,T.MR1             ;CHECK IF MAINT REG 1 CORRECT
031460  001414                          BEQ     45$                     ;YES, CONTINUE
031462  012737  031502  001236          MOV     #43$,$ESCAPE            ;LOAD ESCAPE
031470  012737  044022  001512          MOV     #EMW3,EMW+2             ;LOAD ERROR MESSAGE
031476  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031500  000207                          RTS     PC                      ;MR1 INCORRECT

031502  032777  001000  147430  43$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031510  001036                          BNE     63$                     ;YES, LOOP ON ERROR
031512  042737  002400  002244  45$:    BIC     #MEWD!MCLK,E.MR1        ;SET TRANSISTION
031520  012762  000040  000026          MOV     #DMD,RKMR1(R2)          ;CLOCK TRANSISTION
031526  016237  000026  002204          MOV     RKMR1(R2),T.MR1         ;STORE MR1
031534  023737  002204  002244          CMP     T.MR1,E.MR1             ;CHECK IF MR1 CORRECT
031542  001414                          BEQ     50$                     ;YES, RETURN
031544  012737  031564  001236          MOV     #47$,$ESCAPE            ;LOAD ESCAPE FOR LOOP ON ERROR
031552  012737  044110  001512          MOV     #EMW4,EMW+2             ;LOAD ERROR MESSAGE
031560  011646                          MOV     (SP),-(SP)              ;SAVE RETURN
031562  000207                          RTS     PC                      ;MR1 INCORRECT

031564  032777  001000  147346  47$:    BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031572  001005                          BNE     63$                     ;YES, LOOP ON ERROR
031574  005037  001236          50$:    CLR     $ESCAPE                 ;CLEAR ESCAPE
031600  062716  000002                  ADD     #2,(SP)                 ;ADJUST RETURN
031604  000207                          RTS     PC                      ;RETURN

031606  012706  001100          63$:    MOV     #STACK,SP               ;FORCE STACK
031612  000177  147272                  JMP     @$LPERR                 ;LOOP ON ERROR


                                .SBTTL  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
031616  005737  002320                  RDBIT:  TST     PR.BIT                  ;CHECK IF ONE
031622  001024                                  BNE     10$                     ;YES, SIMULATE ONE
031624  005737  002322                          TST     M1.BIT                  ;CHECK IF PREVIOUS ONE
031630  001404                                  BEQ     4$                      ;NO, INSERT TRANSITION
031632  012762  000440  000026                  MOV     #DMD!MCLK,RKMR1(R2)     ;YES, DO NOT INSERT TRANSITION
031640  000403                                  BR      5$                      ;CLOCK IN ZERO

031642  012762  001440  000026          4$:     MOV     #DMD!MCLK!MERD,RKMR1(R2)        ;INSERT TRANSITION
031650  012762  000040  000026          5$:     MOV     #DMD,RKMR1(R2)          ;CLOCK IN ZERO
031656  012762  000440  000026                  MOV     #DMD!MCLK,RKMR1(R2)
031664  012762  000040  000026                  MOV     #DMD,RKMR1(R2)
031672  000207                                  RTS     PC                      ;RETURN

031674  012762  000440  000026          10$:    MOV     #DMD!MCLK,RKMR1(R2)     ;CLOCK IN ONE
031702  012762  000040  000026                  MOV     #DMD,RKMR1(R2)
031710  012762  001440  000026                  MOV     #DMD!MCLK!MERD,RKMR1(R2)
031716  012762  000040  000026                  MOV     #DMD,RKMR1(R2)
031724  000207                                  RTS     PC                      ;RETURN

                                        .SBTTL  MEMORY CHECK ENABLE TRAP
                                        ;*      IF THE PROCESSOR IN USE HAS MEMORY PARITY OPTION,
                                        ;*      IT WILL BE ENABLED. THIS ROUTINE WILL PROCESS TRAPS CAUSED
                                        ;*      BY MEMORY PARITY ERRORS.
031726  012737  031742  001236          MEMERR: MOV     #10$,$ESCAPE            ;LOAD ESCAPE
031734  011637  002310                          MOV     (SP),TRAPPC             ;STORE PC
031740  104001                                  ERROR   1                       ;REPORT MEM PARITY ERROR
031742  005037  001236                  10$:    CLR     $ESCAPE                 ;CLEAR ESCAPE
031746  032777  001000  147164                  BIT     #SW9,@SWR               ;CHECK IF LOOP ON ERROR
031754  001001                                  BNE     15$                     ;YES, FORCE STACK AND TRY AGAIN
031756  000002                                  RTI                             ;NO, RETURN

031760  012706  001100                  15$:    MOV     #STACK,SP               ;INITIALIZE STACK
031764  000177  147120                          JMP     @$LPERR                 ;LOOP ON ERROR

                                        .SBTTL  CLEAR INPUT BUFFER
031770  012700  046632                  CLRIBF: MOV     #IBUFF,R0               ;GET BUFFER POINTER
031774  010146                                  MOV     R1,-(SP)                ;STORE R1
031776  012701  000400                          MOV     #400,R1 ;SET COUNT FOR CLEAR
032002  005020                          1$:     CLR     (R0)+                   ;CLEAR BUFFER WORD
032004  005301                                  DEC     R1                      ;DEC COUNT
032006  001375                                  BNE     1$                      ;LOOP UNTIL COUNT IS ZERO
032010  012601                                  MOV     (SP)+,R1                ;RESTORE R1
032012  000204                                  RTS     R4                      ;RETURN
                                        .SBTTL  BUILD DATA BUFFER
                                        ;*      THE PATTERN SPECIFIED IN THE CALL IS LOADED INTO THE DATA
                                        ;*      BUFFER. THE ENTIRE BUFFER IS ALWAYS LOADED (400(8) WORDS).
032014                                  BLDDAT:
032014  010046                                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
032016  010146                                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
032020  010346                                  MOV     R3,-(SP)                ;;PUSH R3 ON STACK
032022  010546                                  MOV     R5,-(SP)                ;;PUSH R5 ON STACK
032024  012700  047636                          MOV     #OBUFF,R0               ;LOAD R0 TO POINT TO BUFFER
032030  012701  000400                          MOV     #400,R1                 ;SET DATA LENGTH
032034  021427  000001                          CMP     (R4),#1                 ;TEST IF PATTERN 1(ALL ZERO)
```

```
032040  001004                         BNE     2$                      ;NO - SKIP
032042  005020                 1$:     CLR     (R0)+                   ;ELSE CLEAR OBUFF
032044  005301                         DEC     R1
032046  001375                         BNE     1$
032050  000461                         BR      13$
032052  021427  000007         2$:     CMP     (R4),#7                 ;TEST IF PAT 7 (ALL ONES)
032056  001006                         BNE     4$                      ;NO SKIP
032060  012703  177777                 MOV     #177777,R3              ;ELSE SET BUFFER TO ALL ONES
032064  010320                 3$:     MOV     R3,(R0)+
032066  005301                         DEC     R1
032070  001375                         BNE     3$
032072  000450                         BR      13$
032074  021427  000006         4$:     CMP     (R4),#6                 ;TEST IF PAT 6 (COMPOSIT ROTATING)
032100  001003                         BNE     5$                      ;NO - SKIP
032102  012705  046572                 MOV     #PAT6,R5                ;ELSE GET ADDRESS OF PATTERN 6
032106  000427                         BR      10$                     ;GO LOAD BUFFER
032110  021427  000005         5$:     CMP     (R4),#5                 ;TEST IF PATTERN 5 (ALT 1 & 0)
032114  001006                         BNE     7$                      ;NO SKIP
032116  012703  125252                 MOV     #125252,R3              ;ELSE LOAD OBUFF WITH PAT 5
032122  010320                 6$:     MOV     R3,(R0)+
032124  005301                         DEC     R1
032126  001375                         BNE     6$
032130  000431                         BR      13$
032132  021427  000004         7$:     CMP     (R4),#4                 ;TEST IF PATTERN 4 (HI-LO FREQ MIX)
032136  001003                         BNE     8$                      ;NO - SKIP
032140  012705  046532                 MOV     #PAT4,R5                ;LOAD POINTER WITH ADD OF PAT 4
032144  000410                         BR      10$
032146  021427  000003         8$:     CMP     (R4),#3                 ;TEST IF PAT 3 (MAX PRECOMP PHASE MIX)
032152  001003                         BNE     9$                      ;NO - SKIP
032154  012705  046472                 MOV     #PAT3,R5                ;LOAD POINTER WITH ADD OF PAT 3
032160  000402                         BR      10$                     ;GO LOAD BUFFER
032162  012705  046432         9$:     MOV     #PAT2,R5                ;GET ADDRESS OF PAT 2
032166  012703  000020         10$:    MOV     #16.,R3                 ;SET PATTERN LENGTH COUNT
032172  012520                 11$:    MOV     (R5)+,(R0)+             ;MOV PATTERN INTO OBUFF
032174  005301                         DEC     R1                      ;DEC COUNTERS
032176  005303                         DEC     R3
032200  001374                         BNE     11$                     ;LOOP UNTIL PATTERN IS MOVED
032202  012705  047636                 MOV     #OBUFF,R5               ;SET R5 TO START OF BUFFER
032206  012520                 12$:    MOV     (R5)+,(R0)+             ;REPEAT PATTERN THROUGH BUFFER
032210  005301                         DEC     R1
032212  001375                         BNE     12$
032214  005724                 13$:    TST     (R4)+                   ;BUMP PAST PARAMETER
032216  012605                         MOV     (SP)+,R5                ;;POP STACK INTO R5
032220  012603                         MOV     (SP)+,R3                ;;POP STACK INTO R3
032222  012601                         MOV     (SP)+,R1                ;;POP STACK INTO R1
032224  012600                         MOV     (SP)+,R0                ;;POP STACK INTO R0
032226  000204                         RTS     R4

                               .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

032230  012437  002274                  LOADRK: MOV     (R4)+,L.DCYL            ;LOAD CYLINDER
032234  012437  002266                          MOV     (R4)+,L.DA              ;     TRACK AND SECTOR
032240  012437  002264                          MOV     (R4)+,L.BA              ;     BUS ADDRESS
032244  012437  002262                          MOV     (R4)+,L.WC              ;     WORD COUNT
032250  012737  000040  002276                  MOV     #DMD,L.MR1              ;SET DIAGNOSTIC MODE
032256  005037  002272                          CLR     L.ASOF                  ;CLEAR OFFSET
032262  005037  002270                          CLR     L.CS2                   ;CLEAR CS2
032266  012437  002260                          MOV     (R4)+,L.CS1             ;LOAD CS1
032272  000204                                  RTS     R4                      ;RETURN

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

032274  013762  002262  000002          OPSTRT: MOV     L.WC,RKWC(R2)   ;MOVE THE "L"REGISTERS INTO
032302  013762  002264  000004                  MOV     L.BA,RKBA(R2)   ;CORRESPONDING RK611 REGISTERS.
032310  013762  002274  000020                  MOV     L.DCYL,RKDCYL(R2)
032316  013762  002266  000006                  MOV     L.DA,RKDA(R2)
032324  013762  002270  000010                  MOV     L.CS2,RKCS2(R2)
032332  013762  002276  000026                  MOV     L.MR1,RKMR1(R2)
032340  013762  002272  000016                  MOV     L.ASOF,RKASOF(R2)
032346  013762  002260  000000                  MOV     L.CS1,RKCS1(R2)
032354  000204                                  RTS     R4

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

032356                                  GETREG:
032356  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
032360  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032362  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
032364  010200                                  MOV     R2,R0           ;GET RK BASE
032366  012701  002160                          MOV     #T.CS1,R1       ;GET START OF REGS
032372  012703  000011                          MOV     #11,R3          ;SET COUNT
032376  012021                          1$:     MOV     (R0)+,(R1)+     ;GET CS1 THRU DCYL
032400  005303                                  DEC     R3
032402  001375                                  BNE     1$              ;LOOP UNTIL DONE

032404  062700  000004                          ADD     #4,R0           ;SKIP OVER DB AND SPARE
032410  005021                                  CLR     (R1)+           ;CLEAR T.DB
032412  012021                                  MOV     (R0)+,(R1)+     ;STORE MR1
032414  012037  002212                          MOV     (R0)+,T.ECPS    ;       ECC POSITION
032420  012037  002214                          MOV     (R0)+,T.ECPT    ;       ECC PATTERN
032424  012037  002206                          MOV     (R0)+,T.MR2     ;       MR2
032430  012037  002210                          MOV     (R0)+,T.MR3     ;       MR3
032434  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
032436  012601                                   MOV     (SP)+,R1        ;;POP STACK INTO R1
032440  012600                                   MOV     (SP)+,R0        ;;POP STACK INTO R0
032442  000204                                   RTS     R4

                                         .SBTTL  "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.

032444                                   FSBLVV:
032444  010046                                   MOV     R0,-(SP)        ;;PUSH R0 ON STACK
032446  010146                                   MOV     R1,-(SP)        ;;PUSH R1 ON STACK
032450  010446                                   MOV     R4,-(SP)        ;;PUSH R4 ON STACK
032452  005004                                   CLR     R4              ;CLEAR R4 FOR SWAPPED WORD
032454  012700  000001                           MOV     #BIT0,R0        ;SET FOR BIT TEST
032460  012701  100000                           MOV     #BIT15,R1       ;SET FOR BIT SET
032464  030003                           1$:     BIT     R0,R3           ;TEST IF BIT SET
032466  001401                                   BEQ     2$              ;NO - SKIP
032470  050104                                   BIS     R1,R4           ;SET CORRESPONDING BIT
032472  006300                           2$:     ASL     R0              ;SHIFT FOR NEXT BIT TEST
032474  001403                                   BEQ     3$              ;LAST BIT TESTED - YES - EXIT
032476  000241                                   CLC                     ;CLEAR CARRY
032500  006001                                   ROR     R1              ;SHIFT FOR NEXT BIT SET
032502  000770                                   BR      1$              ;LOOP
032504  010403                           3$:     MOV     R4,R3           ;STORED SWAPPED WORD
032506  012604                                   MOV     (SP)+,R4        ;;POP STACK INTO R4
032510  012601                                   MOV     (SP)+,R1        ;;POP STACK INTO R1
032512  012600                                   MOV     (SP)+,R0        ;;POP STACK INTO R0
032514  000204                                   RTS     R4


                                         .SBTTL  CHECK FOR MEMORY CHECK ENABLE

032516  012737  032602  000004           PARCHK: MOV     #20$,ERRVEC     ;SET VECTOR FOR MEMORY PARITY CHECK
032524  012737  000340  000006                   MOV     #PR7,ERRVEC+2
032532  012737  000000  002334                   MOV     #0,BADPAR       ;LOAD GOOD PARITY
032540  005037  002332                           CLR     MEMPAR          ;CLEAR FLAG
032544  012703  172100                           MOV     #MEMBAS,R3      ;LOAD REGISTER TO DETERMINE IF
                                                                         ;  MEMORY CHECK ENABLE AVAILIBLE
032550  012704  000001                           MOV     #1,R4           ;INITIALIZE MASK
032554  012713  000001                   16$:    MOV     #PAR.EN,(R3)    ;ENABLE MEMORY CHECK
032560  005713                                   TST     (R3)
032562  050437  002332                           BIS     R4,MEMPAR       ;SET FLAG
032566  062703  000002                           ADD     #2,R3
032572  000241                                   CLC
032574  006104                                   ROL     R4              ;CHECK IF FINISHED
032576  001366                                   BNE     16$             ;NO, SET UP NEXT MEMORY PARITY MODULE
032600  000406                                   BR      22$             ;RESTORE TRAP VECTOR
```

```

032602  022626                  20$:    CMP     (SP)+,(SP)+             ;ADJUST STACK
032604  062703  000002                  ADD     #2,R3
032610  000241                          CLC
032612  006104                          ROL     R4                      ;CHECK IF FINISHED
032614  001357                          BNE     16$                     ;NO, GET NEXT LOCATION
032616  012737  000006  000004  22$:    MOV     #ERRVEC+2,ERRVEC        ;RESTORE TRAP CATCHER
032624  005037  000006                  CLR     ERRVEC+2
032630  005737  002332                  TST     MEMPAR                  ;CHECK IF MEMORY CHECK ENABLE AVAILIABLE
032634  001005                          BNE     25$                     ;YES, RETURN
032636  012737  000116  000114          MOV     #MEMVEC+2,MEMVEC        ;RESTORE TRAP CATCHER
032644  005037  000116                  CLR     MEMVEC+2
032650  000207                  25$:    RTS     PC                      ;RETURN
                                .SBTTL  ROUTINE TO SIZE MEMORY


                                ;;*******************************************************************
                                ;*CALL:
                                ;*      JSR     PC,$SIZE
                                ;*      RETURN
                                ;*$LSTAD WILL CONTAIN:
                                ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                ;*      MUST BE SETUP BEFORE THE CALL
                                ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                ;*      DETERMINED BY ROUTINE

032652  010046                  $SIZE:  MOV     R0,-(SP)                ;;SAVE R0 ON THE STACK
032654  010146                          MOV     R1,-(SP)                ;;SAVE R1 ON THE STACK
032656  010246                          MOV     R2,-(SP)                ;;SAVE R2 ON THE STACK
032660  010346                          MOV     R3,-(SP)                ;;SAVE R3 ON THE STACK
032662  013746  000004                  MOV     @#ERRVEC,-(SP)          ;;SAVE PRESENT ERROR VECTOR PS & PC
032666  013746  000006                  MOV     @#ERRVEC+2,-(SP)
032672  010600                          MOV     SP,R0                   ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
032674  104400                          TRAP                            ;;PUSH OLD PSW AND PC ON STACK
032676  012637  000006                  MOV     (SP)+,@#ERRVEC+2          ;;SAVE THE PSW IN @#ERRVEC+2
032702  012701  003776                  MOV     #3776,R1                ;;SETUP ADDRESS
032706  105727                          TSTB    (PC)+                   ;;USE MEMORY MANAGEMENT?
032710  000200                  $KT11:  .WORD   200                     ;;SET TO USE MEMORY MANAGEMENT
032712  100062                          BPL     $CORE                   ;;BR IF NO
032714  012737  033052  000004          MOV     #$KTNEX,@#ERRVEC        ;;SET FOR TIMEOUT
032722  005737  177572                  TST     @#SR0                   ;;KT11 ARE YOU THERE?
032726  052737  100000  032710          BIS     #100000,$KT11           ;;YES--SET KT11 KEY
032734  005046                          CLR     -(SP)                   ;;INITIALIZE FOR "PAR" LOADING
032736  012702  172340                  MOV     #KIPAR0,R2              ;;ADDRESS OF FIRST "PAR"
032742  012703  000010                  MOV     #↑D8,R3                 ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
032746  012762  077406  177740  1$:     MOV     #77406,-40(R2)          ;;PDR = 4K, UP, READ/WRITE
032754  011622                          MOV     (SP),(R2)+              ;;LOAD "PAR"
032756  062716  000200                  ADD     #200,(SP)               ;;UPDATE FOR NEXT "PAR"
032762  077307                          SOB     R3,1$                   ;;LOOP UNTIL ALL EIGHT ARE LOADED
032764  012742  177600                  MOV     #177600,-(R2)           ;;SETUP KIPAR7 FOR I/O
032770  005042                          CLR     -(R2)                   ;;SETUP KIPAR6 FOR TESTING
032772  012737  033010  000004          MOV     #2$,@#ERRVEC            ;;CATCH TIMEOUT IF NO SR3
```

```
033000  012737  000020  172516          MOV     #20,@#SR3               ;;ENABLE 22 BIT MODE
033006  000401                          BR      3$                      ;;THIS PDP-11 HAS A SR3 REGISTER
033010  022626                  2$:     CMP     (SP)+,(SP)+             ;;CLEAN OFF THE STACK--NO SR3
033012  005237  177572          3$:     INC     @#SR0                   ;;TURN ON MEMORY MANAGEMENT
033016  012737  033042  000004          MOV     #$KTOUT,@#ERRVEC        ;;SET FOR TIME OUT
033024  005737  143776          4$:     TST     @#143776                ;;TRAP ON NON-EX-MEM
033030  062712  000040                  ADD     #40,(R2)                ;;MAKE A 1K STEP
033034  023712  172356                  CMP     @#KIPAR7,(R2)           ;;LAST ONE?
033040  101371                          BHI     4$                      ;;NO--TRY IT
033042  011202                  $KTOUT: MOV     (R2),R2                 ;;GET LAST BANK+1
033044  005037  177572                  CLR     @#SR0                   ;;TURN OFF MEMORY MANAGEMENT
033050  000421                          BR      $SIZEX
033052  042737  100000  032710  $KTNEX: BIC     #100000,$KT11           ;;KT11 NON-EXISTENT
033060  012737  033110  000004  $CORE:  MOV     #$CROUT,@#ERRVEC        ;;SET FOR TIMEOUT
033066  005002                          CLR     R2                      ;;SET UP BANK
033070  062701  004000          1$:     ADD     #4000,R1                ;;INCREMENT BY 1K
033074  062702  000040                  ADD     #40,R2                  ;;1K STEP
033100  005711                          TST     (R1)                    ;;TRAP ON TIME OUT
033102  022701  177776                  CMP     #177776,R1              ;;LAST ONE
033106  001370                          BNE     1$                      ;;NO--TRY AGAIN
033110  162701  004000          $CROUT: SUB     #4000,R1
033114  162702  000040          $SIZEX: SUB     #40,R2                  ;;DROP BACK
033120  010006                          MOV     R0,SP                   ;;RESTORE THE STACK
033122  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;RESTORE ERROR VECTOR
033126  012637  000004                  MOV     (SP)+,@#ERRVEC
033132  010137  033154                  MOV     R1,$LSTAD               ;;LAST ADDRESS
033136  010237  033156                  MOV     R2,$LSTBK               ;;LAST BANK
033142  012603                          MOV     (SP)+,R3                ;;RESTORE R3
033144  012602                          MOV     (SP)+,R2                ;;RESTORE R2
033146  012601                          MOV     (SP)+,R1                ;;RESTORE R1
033150  012600                          MOV     (SP)+,R0                ;;RESTORE R0
033152  000207                          RTS     PC
033154  000000                  $LSTAD: .WORD   0                       ;;CONTAINS THE LAST ADDRESS
033156  000000                  $LSTBK: .WORD   0                       ;;CONTAINS THE LAST BANK
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;*************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*       SCOPE                  ;;SCOPE=IOT

033160                          $SCOPE:
033160  104407                          CKSWR                           ;;TEST  FOR CHANGE IN SOFT-SWR
033162  032777  040000  145750  1$:     BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
033170  001131                          BNE     $OVER                   ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
033172  000416                  $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
033174  013746  000004                  MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
```

```
033200  012737  033220  000004            MOV     #5$,@#ERRVEC         ;;SET FOR TIMEOUT
033206  005737  177060                    TST     @#177060             ;;TIME OUT ON XOR?
033212  012637  000004                    MOV     (SP)+,@#ERRVEC       ;;RESTORE THE ERROR VECTOR
033216  000500                            BR      $SVLAD               ;;GO TO THE NEXT TEST
033220  022626                    5$:     CMP     (SP)+,(SP)+          ;;CLEAR THE STACK AFTER A TIME OUT
033222  012637  000004                    MOV     (SP)+,@#ERRVEC       ;;RESTORE THE ERROR VECTOR
033226  000440                            BR      7$                   ;;LOOP ON THE PRESENT TEST
033230                            6$:;#####END OF CODE FOR THE XOR TESTER#####
033230  032777  000400  145702            BIT     #BIT08,@SWR          ;;LOOP ON SPEC. TEST?
033236  001421                            BEQ     2$                   ;;BR IF NO
033240  005046                            CLR     -(SP)                ;;CLEAR A TEMP. LOCATION
033242  117716  145672                    MOVB    @SWR,(SP)            ;;PICKUP THE DESIRED TEST NUMBER
033246  001414                            BEQ     8$                   ;;BRANCH IF BAD TEST NUMBER IN SWR
033250  022716  000036                    CMP     #36,(SP)             ;;CHECK THE NUMBER IN THE SWR
033254  002411                            BLT     8$                   ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
033256  011637  001102                    MOV     (SP),$TSTNM          ;;UPDATE THE TEST NUMBER
033262  005316                            DEC     (SP)                 ;;BACKUP BY ONE
033264  006316                            ASL     (SP)                 ;;SCALE THE TEST NUMBER AS AN INDEX
033266  062716  033472                    ADD     #$SWO8TBL,(SP)       ;;FORM THE ADDRESS OF TEST POINTER
033272  013637  001106                    MOV     @(SP)+,$LPADR        ;;SET LOOP ADDRESS TO DESIRED TEST
033276  000466                            BR      $OVER                ;;GO LOOP ON THE TEST
033300  005726                    8$:     TST     (SP)+                ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
033302  105737  001103            2$:     TSTB    $ERFLG               ;;HAS AN ERROR OCCURRED?
033306  001421                            BEQ     3$                   ;;BR IF NO
033310  123737  001115  001103            CMPB    $ERMAX,$ERFLG        ;;MAX. ERRORS FOR THIS TEST OCCURRED?
033316  101015                            BHI     3$                   ;;BR IF NO
033320  032777  001000  145612            BIT     #BIT09,@SWR          ;;LOOP ON ERROR?
033326  001404                            BEQ     4$                   ;;BR IF NO
033330  013737  001110  001106    7$:     MOV     $LPERR,$LPADR        ;;SET LOOP ADDRESS TO LAST SCOPE
033336  000446                            BR      $OVER
033340  105037  001103            4$:     CLRB    $ERFLG               ;;ZERO THE ERROR FLAG
033344  005037  001234                    CLR     $TIMES               ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
033350  000415                            BR      1$                   ;;ESCAPE TO THE NEXT TEST
033352  032777  004000  145560    3$:     BIT     #BIT11,@SWR          ;;INHIBIT ITERATIONS?
033360  001011                            BNE     1$                   ;;BR IF YES
033362  005737  001256                    TST     $PASS                ;;IF FIRST PASS OF PROGRAM
033366  001406                            BEQ     1$                   ;;       INHIBIT ITERATIONS
033370  005237  001104                    INC     $ICNT                ;;INCREMENT ITERATION COUNT
033374  023737  001234  001104            CMP     $TIMES,$ICNT         ;;CHECK THE NUMBER OF ITERATIONS MADE
033402  002024                            BGE     $OVER                ;;BR IF MORE ITERATION REQUIRED
033404  012737  000001  001104    1$:     MOV     #1,$ICNT             ;;REINITIALIZE THE ITERATION COUNTER
033412  013737  033470  001234            MOV     $MXCNT,$TIMES        ;;SET NUMBER OF ITERATIONS TO DO
033420  105237  001102            $SVLAD: INCB    $TSTNM               ;;COUNT TEST NUMBERS
033424  113737  001102  001254            MOVB    $TSTNM,$TESTN        ;;SET TEST NUMBER IN APT MAILBOX
033432  011637  001106                    MOV     (SP),$LPADR          ;;SAVE SCOPE LOOP ADDRESS
033436  011637  001110                    MOV     (SP),$LPERR          ;;SAVE ERROR LOOP ADDRESS
033442  005037  001236                    CLR     $ESCAPE              ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
033446  112737  000001  001115            MOVB    #1,$ERMAX            ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
033454  013777  001102  145460    $OVER:  MOV     $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER
033462  013716  001106                    MOV     $LPADR,(SP)          ;;FUDGE RETURN ADDRESS
033466  000002                            RTI                          ;;FIXES PS
033470  003720                    $MXCNT: 2000.                        ;;MAX. NUMBER OF ITERATIONS
033472                            $SWO8TBL:
033472  003414                            .WORD   TST1+2               ;;STARTING ADDRESS OF TEST 1
033474  004032                            .WORD   TST2+2               ;;STARTING ADDRESS OF TEST 2
033476  004444                            .WORD   TST3+2               ;;STARTING ADDRESS OF TEST 3
```

```
033500  005056                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
033502  005470                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
033504  006112                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
033506  006362                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
033510  006624                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
033512  007140                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
033514  007432                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
033516  010160                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
033520  010706                          .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
033522  011434                          .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
033524  012162                          .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
033526  012716                          .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
033530  013452                          .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
033532  014200                          .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
033534  014514                          .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
033536  015006                          .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
033540  015534                          .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
033542  016270                          .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
033544  016604                          .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
033546  017266                          .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
033550  017602                          .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
033552  020352                          .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
033554  020662                          .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
033556  021220                          .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
033560  021550                          .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
033562  022104                          .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
033564  022460                          .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
                                ;;************************************************************
                                .SBTTL  LOOP ON INTERNAL ERROR

033566  032777  001000  145344  SCOP1$: BIT     #SW9,@SWR       ;CHECK IF LOOP ON ERROR
033574  001405                          BEQ     5$              ;NO RETURN
033576  105737  001103                  TSTB    $ERFLG          ;CHECK IF ERROR OCCURED
033602  001402                          BEQ     5$              ;NO, RETURN
033604  013716  001110                  MOV     $LPERR,(SP)     ;GO BACK TO BEGINNING OF LOOP
033610  000002                  5$:     RTI                     ;RETURN
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;************************************************************
033612  112737  000001  034056  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
033620  112737  000001  034054  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
033626  000403                          BR      $ATYC
033630  112737  000001  034056  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
033636                          $ATYC:
033636  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
033640  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
033642  105737  034054                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
033646  001450                          BEQ     5$              ;;IF NOT:  BR
033650  122737  000001  001270          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
033656  001031                          BNE     3$              ;;IF NOT:  BR
033660  132737  000100  001271          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
033666  001425                          BEQ     3$              ;;IF NOT:  BR
033670  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
033674  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
033702  005737  001250          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
033706  001375                          BNE     1$              ;;IF NOT:  WAIT
```

```
033710  010037  001264                   MOV     R0,$MSGAD          ;;PUT ADDR IN MAILBOX
033714  105720                   2$:     TSTB    (R0)+              ;;FIND END OF MESSAGE
033716  001376                           BNE     2$
033720  163700  001264                   SUB     $MSGAD,R0          ;;SUB START OF MESSAGE
033724  006200                           ASR     R0                 ;;GET MESSAGE LNGTH IN WORDS
033726  010037  001266                   MOV     R0,$MSGLGT         ;;PUT LENGTH IN MAILBOX
033732  012737  000004  001250           MOV     #4,$MSGTYPE        ;;TELL APT TO TAKE MSG.
033740  000413                           BR      5$
033742  017637  000004  033766   3$:     MOV     @4(SP),4$          ;;PUT MSG ADDR IN JSR LINKAGE
033750  062766  000002  000004           ADD     #2,4(SP)                   ;;BUMP RETURN ADDRESS
033756  013746  177776                   MOV     177776,-(SP)       ;;PUSH 177776 ON STACK
033762  004737  034636                   JSR     PC,$TYPE           ;;CALL TYPE MACRO
033766  000000                   4$:     .WORD   0
033770                           5$:
033770  105737  034056           10$:    TSTB    $FFLG              ;;SHOULD REPORT FATAL ERROR?
033774  001416                           BEQ     12$                ;;IF NOT:  BR
033776  005737  001270                   TST     $ENV               ;;RUNNING UNDER APT?
034002  001413                           BEQ     12$                ;;IF NOT:  BR
034004  005737  001250           11$:    TST     $MSGTYPE           ;;FINISHED LAST MESSAGE?
034010  001375                           BNE     11$                ;;IF NOT:  WAIT
034012  017637  000004  001252           MOV     @4(SP),$FATAL      ;;GET ERROR #
034020  062766  000002  000004           ADD     #2,4(SP)                   ;;BUMP RETURN ADDR.
034026  005237  001250                   INC     $MSGTYPE           ;;TELL APT TO TAKE ERROR
034032  105037  034056           12$:    CLRB    $FFLG              ;;CLEAR FATAL FLAG
034036  105037  034055                   CLRB    $LFLG              ;;CLEAR LOG FLAG
034042  105037  034054                   CLRB    $MFLG              ;;CLEAR MESSAGE FLAG
034046  012601                           MOV     (SP)+,R1           ;;POP STACK INTO R1
034050  012600                           MOV     (SP)+,R0           ;;POP STACK INTO R0
034052  000207                           RTS     PC                 ;;RETURN
034054     000                   $MFLG:  .BYTE   0                  ;;MESSG. FLAG
034055     000                   $LFLG:  .BYTE   0                  ;;LOG FLAG
034056     000                   $FFLG:  .BYTE   0                  ;;FATAL FLAG
        034060                           .EVEN
        000200                   APTSIZE=200
        000001                   APTENV=001
        000100                   APTSPOOL=100
        000040                   APTCSUP=040
                                 .SBTTL  ERROR HANDLER ROUTINE

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

034060                           $ERROR:
034060  104407                           CKSWR                      ;;TEST FOR CHANGE IN SOFT-SWR
034062  105237  001103           7$:     INCB    $ERFLG             ;;SET THE ERROR FLAG
034066  001775                           BEQ     7$                 ;;DON'T LET THE FLAG GO TO ZERO
034070  013777  001102  145044           MOV     $TSTNM,@DISPLAY    ;;DISPLAY TEST NUMBER AND ERROR FLAG
```

```
034076  032777  002000  145034           BIT     #BIT10,@SWR      ;;BELL ON ERROR?
034104  001402                           BEQ     1$               ;;NO - SKIP
034106  104401  001240                   TYPE    ,$BELL           ;;RING BELL
034112  005237  001112           1$:     INC     $ERTTL           ;;COUNT THE NUMBER OF ERRORS
034116  011637  001116                   MOV     (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
034122  162737  000002  001116           SUB     #2,$ERRPC
034130  117737  144762  001114           MOVB    @$ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
034136  032777  020000  144774           BIT     #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
034144  001004                           BNE     20$              ;;SKIP TYPEOUTS
034146  004737  034260                   JSR     PC,TYPERR        ;;GO TO USER ERROR ROUTINE
034152  104401  001245                   TYPE    ,$CRLF
034156                           20$:
034156  122737  000001  001270           CMPB    #APTENV,$ENV     ;;RUNNING IN APT MODE
034164  001007                           BNE     2$               ;;NO,SKIP APT ERROR REPORT
034166  113737  001114  034200           MOVB    $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
034174  004737  033630                   JSR     PC,$ATY4         ;;REPORT FATAL ERROR TO APT
034200     000                   21$:    .BYTE   0
034201     000                           .BYTE   0
034202  000777                   22$:    BR      22$              ;;APT ERROR LOOP
034204  005777  144730           2$:     TST     @SWR             ;;HALT ON ERROR
034210  100002                           BPL     3$               ;;SKIP IF CONTINUE
034212  000000                           HALT                     ;;HALT ON ERROR!
034214  104407                           CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
034216  032777  001000  144714   3$:     BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
034224  001402                           BEQ     4$               ;;BR IF NO
034226  013716  001110                   MOV     $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
034232  005737  001236           4$:     TST     $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
034236  001402                           BEQ     5$               ;;BR IF NONE
034240  013716  001236                   MOV     $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
034244                           5$:
034244  022737  023276  000042           CMP     #$ENDAD,@#42     ;;ACT-11 AUTO-ACCEPT?
034252  001001                           BNE     6$               ;;BRANCH IF NO
034254  000000                           HALT                     ;;YES
034256                           6$:
034256  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.
                                 ;;*************************************************************
034260  104413                   TYPERR: SAVREG
034262  113737  001102  001254           MOVB    $TSTNM,$TESTN    ;SAVE TEST NUMBER FOR REPORT
034270  042737  177400  001254           BIC     #177400,$TESTN   ;CLEAR UNUSED BITS
034276  113700  001114                   MOVB    $ITEMB,R0        ;ENTER ERROR NUMBER
034302  042700  177400                   BIC     #177400,R0       ;CLEAR UNUSED BITS
034306  005300                           DEC     R0               ;FORM INDEX FOR ERROR TABLE
034310  006300                           ASL     R0
034312  006300                           ASL     R0
034314  006300                           ASL     R0
034316  062700  001330           1$:     ADD     #$ERRTB,R0       ;FORM ADDRESS OF ERROR ENTRY
```

```
034322  012037  034336                  MOV     (R0)+,2$        ;GET EM POINTER
034326  001404                          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
034330  104401  001245                  TYPE    ,$CRLF          ;TYPE CARRIAGE RETURN LINE FEED
034334  104401                          TYPE                    ;TYPE ERROR MESSAGE (EM)
034336  000000                  2$:     .WORD   0               ;EM POINTER GOES HERE
034340  012037  034354          3$:     MOV     (R0)+,4$        ;GET DH POINTER
034344  001404                          BEQ     5$              ;BRANCH IF THERE ISN'T ONE
034346  104401  001245                  TYPE    ,$CRLF          ;TYPE CR-LF
034352  104401                          TYPE                    ;TYPE DATA HEADER
034354  000000                  4$:     .WORD   0               ;DH POINTER GOES HERE
034356  012001                  5$:     MOV     (R0)+,R1        ;GET DT POINTER
034360  001445                          BEQ     20$             ;BRANCH IF THERE ARE NONE
034362  005004                          CLR     R4              ;RESET INDENT SWITCH
034364  012000                          MOV     (R0)+,R0        ;GET DF POINTER
034366  012002                          MOV     (R0)+,R2        ;STORE NUMBER OF DH'S
034370  104401  001245                  TYPE    ,$CRLF
034374  112003                  10$:    MOVB    (R0)+,R3        ;GET & STORE NUMBER OF DATA WORDS
034376  105720                          TSTB    (R0)+           ;BUMP PAST FORMAT WORD
034400  005703                          TST     R3              ;TEST IF ANY DATA FOR THIS HEADER
034402  001416                          BEQ     14$             ;NO - SKIP DATA PRINT
034404  005704                          TST     R4              ;CHECK IF INDENT WORDS
034406  001004                          BNE     12$             ;YES, GO INDENT
034410  013146                  11$:    MOV     @(R1)+,-(SP)    ;PUT FIRST DATA WORD ON STACK
034412  104402                          TYPOC                   ;TYPE IT
034414  005303                          DEC     R3              ;MORE DATA WORDS
034416  001403                          BEQ     13$             ;NO-BRANCH
034420  104401  041116          12$:    TYPE    ,SPACE2         ;TYPE SEPARATORS
034424  000771                          BR      11$             ;LOOP
034426  104401  001245          13$:    TYPE    ,$CRLF          ;TYPE <CR><LF>
034432  005710                          TST     (R0)            ;CHECK IF NEXT HEADER AVAILIBLE
034434  001401                          BEQ     14$             ;NO, DO NOT CHANGE INDENT
034436  005104                          COM     R4              ;CHANGE INDENT
034440  005302                  14$:    DEC     R2              ;MORE DH'S?
034442  003414                          BLE     20$             ;NO-BRANCH
034444  012037  034464          15$:    MOV     (R0)+,18$       ;GET NEXT DH POINTER
034450  001751                          BEQ     10$             ;IF NO HEADER GET DATA
034452  005704                          TST     R4              ;INDENT?
034454  001402                          BEQ     17$             ;NO-BRANCH
034456  104401  041116                  TYPE    ,SPACE2         ;INDENT
034462  104401                  17$:    TYPE                    ;TYPE DH
034464  000000                  18$:    .WORD   0               ;DH POINTER GOES HERE
034466  104401  001245                  TYPE    ,$CRLF
034472  000740                          BR      10$             ;LOOP
034474  104414                  20$:    RESREG
034476  005237  002314                  INC     ERRCNT          ;INCREMENT ERROR COUNT
034502  032777  010000  144430          BIT     #SW12,@SWR      ;CHECK IF ABORT AFTER 20 ERRORS
034510  001421                          BEQ     25$             ;NO, RETURN
034512  022737  000024  002314          CMP     #20.,ERRCNT     ;CHECK IF EROR THRESHOLD EXCEEDED
034520  103015                          BHIS    25$             ;NO, RETURN
034522  104401  041121                  TYPE    ,ABORT          ;TYPE "PROGRAM HAS BEEN ABORTED BECAUSE
                                                                ;  ERROR THRESHOLD EXCEEDED"
034526  005737  000042                  TST     42              ;CHECK IF IN CHAIN MODE
034532  001407                          BEQ     30$             ;NO, HALT
034534  012737  000001  023134          MOV     #1,$EOPCT       ;FORCE END OF PASS COUNT TO ONE FOR ABORT
034542  012706  001100                  MOV     #STACK,SP       ;INITIALIZE STACK
034546  000137  023106                  JMP     $EOP            ;BRING IN NEXT PROGRAM IN CHAIN
```

```
034552  000000                          30$:    HALT
034554  000207                          25$:    RTS     PC

                                        .SBTTL  CONTROLLED PROGRAM HALT

034556  013702  001324                  CTRHLT: MOV     $BASE,R2                ;SET '611 BASE
034562  012762  005000  000010                  MOV     #CLR,RKCS2(R2)          ;CLEAR SUBSYSTEM
034570  005037  001236                          CLR     $ESCAPE                 ;CLEAR ESCAPE
034574  105037  001103                          CLRB    $ERFLG                  ;CLEAR ERROR FLAG
034600  012706  001100                          MOV     #STACK,SP               ;CLEAR STACK
034604  104401  041053                          TYPE    ,OPR007                 ;TYPE HALT MESSAGE
034610  005737  000042                          TST     42                      ;TEST IF MONITOR PRESENT
034614  001405                                  BEQ     5$                      ;NO SKIP
034616  012737  000001  023134                  MOV     #1,$EOPCT               ;FORCE END OF PROGRAM
034624  000137  023106                          JMP     $EOP                    ;JUMP TO END OF PASS

034630  000000                          5$:     HALT                            ;HALT PROGRAM
034632  000137  002404                          JMP     START1                  ;RESTART IF CONTINUE

                                        .SBTTL  TYPE ROUTINE

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

034636  105737  001157                  $TYPE:  TSTB    $TPFLG                  ;;IS THERE A TERMINAL?
034642  100002                                  BPL     1$                      ;;BR IF YES
034644  000000                                  HALT                            ;;HALT HERE IF NO TERMINAL
034646  000430                                  BR      3$                      ;;LEAVE
034650  010046                          1$:     MOV     R0,-(SP)                ;;SAVE R0
034652  017600  000002                          MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
034656  122737  000001  001270                  CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
034664  001011                                  BNE     62$                     ;;NO,GO CHECK FOR APT CONSOLE
034666  132737  000100  001271                  BITB    #APTSPOOL,$ENVM         ;;SPOOL MESSAGE TO APT
034674  001405                                  BEQ     62$                     ;;NO,GO CHECK FOR CONSOLE
034676  010037  034706                          MOV     R0,61$                  ;;SETUP MESSAGE ADDRESS FOR APT
034702  004737  033620                          JSR     PC,$ATY3                ;;SPOOL MESSAGE TO APT
034706  000000                          61$:    .WORD   0                       ;;MESSAGE ADDRESS
034710  132737  000040  001271          62$:    BITB    #APTCSUP,$ENVM          ;;APT CONSOLE SUPPRESSED
034716  001003                                  BNE     60$                     ;;YES,SKIP TYPE OUT
034720  112046                          2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
034722  001005                                  BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
034724  005726                                  TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
034726  012600                          60$:    MOV     (SP)+,R0                ;;RESTORE R0
034730  062716  000002                  3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
```

```
034734  000002                                    RTI                               ;;RETURN
034736  122716  000011                  4$:       CMPB     #HT,(SP)                 ;;BRANCH IF <HT>
034742  001430                                    BEQ      8$
034744  122716  000200                            CMPB     #CRLF,(SP)               ;;BRANCH IF NOT <CRLF>
034750  001006                                    BNE      5$
034752  005726                                    TST      (SP)+                    ;;POP  <CR><LF> EQUIV
034754  104401                                    TYPE                              ;;TYPE A CR AND LF
034756  001245                                    $CRLF
034760  105037  035114                            CLRB     $CHARCNT                 ;;CLEAR CHARACTER COUNT
034764  000755                                    BR       2$                       ;;GET NEXT CHARACTER
034766  004737  035050                  5$:       JSR      PC,$TYPEC                ;;GO TYPE THIS CHARACTER
034772  123726  001156                  6$:       CMPB     $FILLC,(SP)+             ;;IS IT TIME FOR FILLER CHARS.?
034776  001350                                    BNE      2$                       ;;IF NO GO GET NEXT CHAR.
035000  013746  001154                            MOV      $NULL,-(SP)              ;;GET # OF FILLER CHARS. NEEDED
                                                                                    ;;AND THE NULL CHAR.
035004  105366  000001                  7$:       DECB     1(SP)                    ;;DOES A NULL NEED TO BE TYPED?
035010  002770                                    BLT      6$                       ;;BR IF NO--GO POP THE NULL OFF OF STACK
035012  004737  035050                            JSR      PC,$TYPEC                ;;GO TYPE A NULL
035016  105337  035114                            DECB     $CHARCNT                 ;;DO NOT COUNT AS A COUNT
035022  000770                                    BR       7$                       ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

035024  112716  000040                  8$:       MOVB     #' ,(SP)                 ;;REPLACE TAB WITH SPACE
035030  004737  035050                  9$:       JSR      PC,$TYPEC                ;;TYPE A SPACE
035034  132737  000007  035114                    BITB     #7,$CHARCNT              ;;BRANCH IF NOT AT
035042  001372                                    BNE      9$                       ;;TAB STOP
035044  005726                                    TST      (SP)+                    ;;POP SPACE OFF STACK
035046  000724                                    BR       2$                       ;;GET NEXT CHARACTER
035050  105777  144074                  $TYPEC:   TSTB     @$TPS                    ;;WAIT UNTIL PRINTER IS READY
035054  100375                                    BPL      $TYPEC
035056  116677  000002  144066                    MOVB     2(SP),@$TPB              ;;LOAD CHAR TO BE TYPED INTO DATA REG.
035064  122766  000015  000002                    CMPB     #CR,2(SP)                ;;IS CHARACTER A CARRIAGE RETURN?
035072  001003                                    BNE      1$                       ;;BRANCH IF NO
035074  105037  035114                            CLRB     $CHARCNT                 ;;YES--CLEAR CHARACTER COUNT
035100  000406                                    BR       $TYPEX                   ;;EXIT
035102  122766  000012  000002          1$:       CMPB     #LF,2(SP)                ;;IS CHARACTER A LINE FEED?
035110  001402                                    BEQ      $TYPEX                   ;;BRANCH IF YES
035112  105227                                    INCB     (PC)+                    ;;COUNT THE CHARACTER
035114  000000                          $CHARCNT:.WORD     0                        ;;CHARACTER COUNT STORAGE
035116  000207                          $TYPEX:   RTS      PC

                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

```
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPON                           ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                           ;;CALL FOR TYPEOUT

035120  017646  000000          $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
035124  116637  000001  035343          MOVB    1(SP),$0FILL            ;;LOAD ZERO FILL SWITCH
035132  112637  035345                  MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
035136  062716  000002                  ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
035142  000406                          BR      $TYPON
035144  112737  000001  035343  $TYPOC: MOVB    #1,$0FILL               ;;SET THE ZERO FILL SWITCH
035152  112737  000006  035345          MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
035160  112737  000005  035342  $TYPON: MOVB    #5,$0CNT                ;;SET THE ITERATION COUNT
035166  010346                          MOV     R3,-(SP)                ;;SAVE R3
035170  010446                          MOV     R4,-(SP)                ;;SAVE R4
035172  010546                          MOV     R5,-(SP)                ;;SAVE R5
035174  113704  035345                  MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
035200  005404                          NEG     R4
035202  062704  000006                  ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
035206  110437  035344                  MOVB    R4,$OMODE               ;;SAVE IT FOR USE
035212  113704  035343                  MOVB    $0FILL,R4               ;;GET THE ZERO FILL SWITCH
035216  016605  000012                  MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
035222  005003                          CLR     R3                      ;;CLEAR THE OUTPUT WORD
035224  006105                  1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
035226  000404                          BR      3$                      ;;GO DO MSB
035230  006105                  2$:     ROL     R5                      ;;FORM THIS DIGIT
035232  006105                          ROL     R5
035234  006105                          ROL     R5
035236  010503                          MOV     R5,R3
035240  006103                  3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
035242  105337  035344                  DECB    $OMODE                  ;;TYPE THIS DIGIT?
035246  100016                          BPL     7$                      ;;BR IF NO
035250  042703  177770                  BIC     #177770,R3              ;;GET RID OF JUNK
035254  001002                          BNE     4$                      ;;TEST FOR 0
035256  005704                          TST     R4                      ;;SUPPRESS THIS 0?
035260  001403                          BEQ     5$                      ;;BR IF YES
035262  005204                  4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
035264  052703  000060                  BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
035270  052703  000040          5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
035274  110337  035340                  MOVB    R3,8$                   ;;SAVE FOR TYPING
035300  104401  035340                  TYPE    ,8$                     ;;GO TYPE THIS DIGIT
035304  105337  035342          7$:     DECB    $0CNT                   ;;COUNT BY 1
035310  003347                          BGT     2$                      ;;BR IF MORE TO DO
035312  002402                          BLT     6$                      ;;BR IF DONE
035314  005204                          INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
035316  000744                          BR      2$                      ;;GO DO THE LAST DIGIT
035320  012605                  6$:     MOV     (SP)+,R5                ;;RESTORE R5
035322  012604                          MOV     (SP)+,R4                ;;RESTORE R4
035324  012603                          MOV     (SP)+,R3                ;;RESTORE R3
035326  016666  000002  000004          MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
```

```
035334  012616                          MOV     (SP)+,(SP)
035336  000002                          RTI                             ;;RETURN
035340     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
035341     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
035342     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
035343     000                  $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
035344  000000                  $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

035346                          $TYPDS:
035346  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
035350  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
035352  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
035354  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
035356  010546                          MOV     R5,-(SP)                ;;PUSH R5 ON STACK
035360  012746  020200                  MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
035364  016605  000020                  MOV     20(SP),R5               ;;GET THE INPUT NUMBER
035370  100004                          BPL     1$                      ;;BR IF INPUT IS POS.
035372  005405                          NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
035374  112766  000055  000001          MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
035402  005000                  1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
035404  012703  035562                  MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
035410  112723  000040                  MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
035414  005002                  2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
035416  016001  035552                  MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
035422  160105                  3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
035424  002402                          BLT     4$                      ;;BR IF DONE
035426  005202                          INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
035430  000774                          BR      3$
035432  060105                  4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
035434  005702                          TST     R2                      ;;CHECK IF BCD DIGIT=0
035436  001002                          BNE     5$                      ;;FALL THROUGH IF 0
035440  105716                          TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
035442  100407                          BMI     7$                      ;;BR IF YES
035444  106316                  5$:     ASLB    (SP)                    ;;MSD?
035446  103003                          BCC     6$                      ;;BR IF NO
035450  116663  000001  177777          MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
035456  052702  000060          6$:     BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
035462  052702  000040          7$:     BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
035466  110223                          MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
035470  005720                          TST     (R0)+                   ;;JUST INCREMENTING
035472  020027  000010                  CMP     R0,#10                  ;;CHECK THE TABLE INDEX
035476  002746                          BLT     2$                      ;;GO DO THE NEXT DIGIT
035500  003002                          BGT     8$                      ;;GO TO EXIT
035502  010502                          MOV     R5,R2                   ;;GET THE LSD
035504  000764                          BR      6$                      ;;GO CHANGE TO ASCII
```

```
035506  105726                  8$:     TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
035510  100003                          BPL     9$                      ;;BR IF NO
035512  116663  177777  177776          MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
035520  105013                  9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
035522  012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
035524  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
035526  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
035530  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
035532  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
035534  104401  035562                  TYPE    ,$DBLK                  ;;NOW TYPE THE NUMBER
035540  016666  000002  000004          MOV     2(SP),4(SP)             ;;ADJUST THE STACK
035546  012616                          MOV     (SP)+,(SP)
035550  000002                          RTI                             ;;RETURN TO USER
035552  023420                  $DTBL:  10000.
035554  001750                          1000.
035556  000144                          100.
035560  000012                          10.
035562  000004                  $DBLK:  .BLKW   4
                                .SBTTL  TTY INPUT ROUTINE


                                ;;************************************************************
                                .ENABL  LSB
035572  000000                  $TKCNT:  .WORD  0                       ;;NUMBER OF ITEMS IN QUEUE
035574  000000                  $TKQIN:  .WORD  0                       ;;INPUT POINTER
035576  000000                  $TKQOUT: .WORD  0                       ;;OUTPUT POINTER
035600  000001                  $TKQSRT: .BLKB  1                       ;;TTY KEYBOARD QUEUE
        035601                  $TKQEND=.
        035602                  .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*       JSR     PC,$TKINT
                                ;*       RETURN
                                ;
035602  005037  035572          $TKINT: CLR     $TKCNT                  ;;CLEAR COUNT OF ITEMS IN QUEUE
035606  012737  035600  035574          MOV     #$TKQSRT,$TKQIN         ;;MOVE THE STARTING ADDRESS OF THE
035614  013737  035574  035576          MOV     $TKQIN,$TKQOUT          ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
035622  012737  035652  000060          MOV     #$TKSRV,@#TKVEC         ;;INITIALIZE THE KEYBOARD VECTOR
035630  012737  000200  000062          MOV     #200,@#TKVEC+2          ;;"BR" LEVEL 4
035636  005777  143304                  TST     @$TKB                   ;;CLEAR DONE FLAG
035642  012777  000100  143274          MOV     #100,@$TKS              ;;ENABLE TTY KEYBOARD INTERRUPT
035650  000207                          RTS     PC                      ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRHLT)
                                ;
035652  117746  143270          $TKSRV: MOVB    @$TKB,-(SP)             ;;PICKUP THE CHARACTER
035656  042716  177600                  BIC     #↑C177,(SP)             ;;STRIP THE JUNK
035662  021627  000003                  CMP     (SP),#3                 ;;IS IT A CONTROL C?
```

```
035666  001007                              BNE     1$                  ;;BRANCH IF NO
035670  104401  036766                      TYPE    ,$CNTLC             ;;TYPE A CONTROL-C (↑C)
035674  004737  035602                      JSR     PC,$TKINT           ;;INIT THE KEYBOARD
035700  005726                              TST     (SP)+               ;;CLEAN UP STACK
035702  000137  034556                      JMP     CTRHLT              ;;CONTROL C RESTART
035706  021627  000007              1$:     CMP     (SP),#7             ;;IS IT A CONTROL G?
035712  001004                              BNE     2$                  ;;BRANCH IF NO
035714  022737  000176  001140              CMP     #SWREG,SWR          ;;IS SOFT-SWR SELECTED?
035722  001500                              BEQ     6$                  ;;GO TO SWR CHANGE

035724                              2$:
035724  022737  000001  035572              CMP     #1,$TKCNT           ;;IS THE QUEUE FULL?
035732  001004                              BNE     3$                  ;;BRANCH IF NO
035734  104401  001240                      TYPE    ,$BELL              ;;RING THE TTY BELL
035740  005726                              TST     (SP)+               ;;CLEAN CHARACTER OFF OF STACK
035742  000451                              BR      5$                  ;;EXIT
035744  021627  000023              3$:     CMP     (SP),#23            ;;IS IT A CONTROL-S?
035750  001021                              BNE     32$                 ;;BRANCH IF NO
035752  005077  143166                      CLR     @$TKS               ;;DISABLE TTY KEYBOARD INTERRUPTS
035756  005726                              TST     (SP)+               ;;CLEAN CHAR OFF STACK
035760  105777  143160              31$:    TSTB    @$TKS               ;;WAIT FOR A CHAR
035764  100375                              BPL     31$                 ;;LOOP UNTIL ITS THERE
035766  117746  143154                      MOVB    @$TKB,-(SP)         ;;GET THE CHARACTER
035772  042716  177600                      BIC     #↑C177,(SP)         ;;MAKE IT 7-BIT ASCII
035776  022627  000021                      CMP     (SP)+,#21           ;;IS IT A CONTROL-Q?
036002  001366                              BNE     31$                 ;;BRANCH IF NO
036004  012777  000100  143132              MOV     #100,@$TKS          ;;REENABLE TTY KEYBOARD INTERRUPTS
036012  000002                              RTI                         ;;RETURN
036014  005237  035572              32$:    INC     $TKCNT              ;;COUNT THIS CHARACTER
036020  021627  000140                      CMP     (SP),#140           ;;IS IT UPPER CASE?
036024  002405                              BLT     4$                  ;;BRANCH IF YES
036026  021627  000175                      CMP     (SP),#175           ;;IS IT A SPECIAL CHAR?
036032  003002                              BGT     4$                  ;;BRANCH IF YES
036034  042716  000040                      BIC     #40,(SP)            ;;MAKE IT UPPER CASE
036040  112677  177530              4$:     MOVB    (SP)+,@$TKQIN       ;;AND PUT IT IN QUEUE
036044  005237  035574                      INC     $TKQIN              ;;UPDATE THE POINTER
036050  023727  035574  035601              CMP     $TKQIN,#$TKQEND     ;;GO OFF THE END?
036056  001003                              BNE     5$                  ;;BRANCH IF NO
036060  012737  035600  035574              MOV     #$TKQSRT,$TKQIN     ;;RESET THE POINTER
036066  000002                      5$:     RTI                         ;;RETURN

                                    ;;*******************************************************************
                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                    ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                    ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
036070  022737  000176  001140      $CKSWR: CMP     #SWREG,SWR          ;;IS THE SOFT-SWR SELECTED
036076  001124                              BNE     15$                 ;;EXIT IF NOT
036100  105777  143040                      TSTB    @$TKS               ;;IS A CHAR WAITING?
036104  100121                              BPL     15$                 ;;IF NOT, EXIT
036106  117746  143034                      MOVB    @$TKB,-(SP)         ;;YES
036112  042716  177600                      BIC     #↑C177,(SP)         ;;MAKE IT  7-BIT ASCII
036116  021627  000007                      CMP     (SP),#7             ;;IS IT A CONTROL-G?
036122  001300                              BNE     2$                  ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                        ;;AND EXIT
```

```
                                        ;;*******************************************************************
                                        ;;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
036124  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
036132  001674                          BEQ     2$              ;;BRANCH IF YES
036134  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
036136  004737  035602                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
036142  005077  142776                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
036146  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

036154  104401  037000                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (↑G)
036160  104401  037005          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
036164  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
036170  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
036172  104401  037016                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
036176  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
036200  005046                          CLR     -(SP)           ;;THE NEW SWR
036202  105777  142736          7$:     TSTB    @$TKS           ;;CHAR THERE?
036206  100375                          BPL     7$              ;;IF NOT TRY AGAIN

036210  117746  142732                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
036214  042716  177600                  BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII

036220  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
036224  001015                          BNE     9$              ;;BRANCH IF NOT
036226  104401  036766                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
036232  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
036236  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
036244  001003                          BNE     8$              ;;BRANCH IF NO
036246  012777  000100  142670          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
036254  000137  034556          8$:     JMP     CTRHLT          ;;CONTROL-C RESTART


036260  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
036264  001005                          BNE     10$             ;;BRANCH IF NOT
036266  104401  036773                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
036272  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
036276  000737                          BR      19$             ;;LET'S TRY IT AGAIN


036300  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
036304  001022                          BNE     16$             ;;BRANCH IF NO
036306  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
036312  001403                          BEQ     11$             ;;BRANCH IF YES
036314  016677  000002  142616          MOV     2(SP),@SWR      ;;SAVE NEW SWR
036322  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
036326  104401  001245          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
036332  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
036340  001003                          BNE     15$             ;;BRANCH IF NOT
036342  012777  000100  142574          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
036350  000002                  15$:    RTI                     ;;RETURN
036352  004737  035050          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
036356  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
036362  002420                          BLT     18$             ;;BRANCH IF YES
036364  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
```

```
036370  003015                          BGT     18$                     ;;BRANCH IF YES
036372  042726  000060                  BIC     #60,(SP)+               ;;STRIP-OFF ASCII
036376  005766  000002                  TST     2(SP)                   ;;IS THIS THE FIRST CHAR
036402  001403                          BEQ     17$                     ;;BRANCH IF YES
036404  006316                          ASL     (SP)                    ;;NO, SHIFT PRESENT
036406  006316                          ASL     (SP)                    ;;    CHAR OVER TO MAKE
036410  006316                          ASL     (SP)                    ;;    ROOM FOR NEW ONE.
036412  005266  000002          17$:    INC     2(SP)                   ;;KEEP COUNT OF CHAR
036416  056616  177776                  BIS     -2(SP),(SP)             ;;SET IN NEW CHAR
036422  000667                          BR      7$                      ;;GET THE NEXT ONE
036424  104401  001244          18$:    TYPE    ,$QUES                  ;;TYPE ?<CR><LF>
036430  000720                          BR      20$                     ;;SIMULATE CONTROL-U
                                .DSABL  LSB


                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                           ;;GET A CHARACTER FROM THE QUEUE
                                ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
                                ;*                                      ;;WITH PARITY BIT STRIPPED OFF
                                ;

036432  011646                  $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC AND
036434  016666  000004  000002          MOV     4(SP),2(SP)             ;;THE PS
036442  005066  000004                  CLR     4(SP)                   ;;GET READY FOR A CHARACTER
036446  005046                          CLR     -(SP)                   ;;PUT NEW PS ON STACK
036450  012746  036456                  MOV     #64$,-(SP)              ;;PUT NEW PC ON  STACK
036454  000002                          RTI                             ;;POP NEW PC AND PS
036456                          64$:
036456  005737  035572          1$:     TST     $TKCNT                  ;;WAIT ON A CHARACTER
036462  001775                          BEQ     1$
036464  005337  035572                  DEC     $TKCNT                  ;;DECREMENT THE COUNTER
036470  117766  177102  000004          MOVB    @$TKQOUT,4(SP)          ;;GET ONE CHARACTER
036476  005237  035576                  INC     $TKQOUT                 ;;UPDATE THE POINTER
036502  023727  035576  035601          CMP     $TKQOUT,#$TKQEND        ;;DID IT GO OFF OF THE END?
036510  001003                          BNE     2$                      ;;BRANCH IF NO
036512  012737  035600  035576          MOV     #$TKQSRT,$TKQOUT        ;;RESET THE POINTER
036520  000002                  2$:     RTI                             ;;RETURN
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

036522  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
036524  005046                          CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
036526  012703  036756          1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
036532  022703  036766          2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
036536  101456                          BLOS    4$                      ;;BR IF YES
036540  104410                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
036542  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
036544  122713  000177          10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
036550  001022                          BNE     5$                      ;;BR IF NO
036552  005716                          TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
```

```
036554  001007                                   BNE     6$                      ;;BR IF NO
036556  112737  000134  036754                   MOVB    #'\,9$                  ;;TYPE A BACK SLASH
036564  104401  036754                           TYPE    ,9$
036570  012716  177777                           MOV     #-1,(SP)                ;;SET THE RUBOUT KEY
036574  005303                          6$:      DEC     R3                      ;;BACKUP BY ONE
036576  020327  036756                           CMP     R3,#$TTYIN              ;;STACK EMPTY?
036602  103434                                   BLO     4$                      ;;BR IF YES
036604  111337  036754                           MOVB    (R3),9$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
036610  104401  036754                           TYPE    ,9$                     ;;GO TYPE
036614  000746                                   BR      2$                      ;;GO READ ANOTHER CHAR.
036616  005716                          5$:      TST     (SP)                    ;;RUBOUT KEY SET?
036620  001406                                   BEQ     7$                      ;;BR IF NO
036622  112737  000134  036754                   MOVB    #'\,9$                  ;;TYPE A BACK SLASH
036630  104401  036754                           TYPE    ,9$
036634  005016                                   CLR     (SP)                    ;;CLEAR THE RUBOUT KEY
036636  122713  000025                  7$:      CMPB    #25,(R3)                ;;IS CHARACTER A CTRL U?
036642  001003                                   BNE     8$                      ;;BR IF NO
036644  104401  036773                           TYPE    ,$CNTLU                 ;;TYPE A CONTROL "U"
036650  000726                                   BR      1$                      ;;GO START OVER
036652  122713  000022                  8$:      CMPB    #22,(R3)                ;;IS CHARACTER A "↑R"?
036656  001011                                   BNE     3$                      ;;BRANCH IF NO
036660  105013                                   CLRB    (R3)                    ;;CLEAR THE CHARACTER
036662  104401  001245                           TYPE    ,$CRLF                  ;;TYPE A "CR" & "LF"
036666  104401  036756                           TYPE    ,$TTYIN                 ;;TYPE THE INPUT STRING
036672  000717                                   BR      2$                      ;;GO PICKUP ANOTHER CHACTER
036674  104401  001244                  4$:      TYPE    ,$QUES                  ;;TYPE A '?'
036700  000712                                   BR      1$                      ;;CLEAR THE BUFFER AND LOOP
036702  111337  036754                  3$:      MOVB    (R3),9$                 ;;ECHO THE CHARACTER
036706  104401  036754                           TYPE    ,9$
036712  122723  000015                           CMPB    #15,(R3)+               ;;CHECK FOR RETURN
036716  001305                                   BNE     2$                      ;;LOOP IF NOT RETURN
036720  105063  177777                           CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
036724  104401  001246                           TYPE    ,$LF                    ;;TYPE A LINE FEED
036730  005726                                   TST     (SP)+                   ;;CLEAN RUBOUT KEY FROM THE STACK
036732  012603                                   MOV     (SP)+,R3                ;;RESTORE R3
036734  011646                                   MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
036736  016666  000004  000002                   MOV     4(SP),2(SP)             ;;       FIRST ASCII CHARACTER ON IT
036744  012766  036756  000004                   MOV     #$TTYIN,4(SP)
036752  000002                                   RTI                             ;;RETURN
036754     000                          9$:      .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
036755     000                                   .BYTE   0                       ;;TERMINATOR
036756  000010                          $TTYIN:  .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
036766  041536  005015     000          $CNTLC:  .ASCIZ  /↑C/<15><12>            ;;CONTROL "C"
036773     136  006525  000012          $CNTLU:  .ASCIZ  /↑U/<15><12>            ;;CONTROL "U"
037000  043536  005015     000          $CNTLG:  .ASCIZ  /↑G/<15><12>            ;;CONTROL "G"
037005     015  051412  051127          $MSWR:   .ASCIZ  <15><12>/SWR = /
037012  036440  000040
037016  020040  042516  020127          $MNEW:   .ASCIZ  /  NEW = /
037024  020075     000
        037030                          .EVEN
                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY


                                        ;;*****************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
```

```
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                       ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE                 ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                                  ;;HIGH ORDER BITS ARE IN $HIOCT

037030  011646                  $RDOCT: MOV     (SP),-(SP)          ;;PROVIDE SPACE FOR THE
037032  016666  000004  000002          MOV     4(SP),2(SP)         ;;INPUT NUMBER
037040  010046                          MOV     R0,-(SP)            ;;PUSH R0 ON STACK
037042  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
037044  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
037046  104411                  1$:     RDLIN                       ;;READ AN ASCIZ LINE
037050  012600                          MOV     (SP)+,R0            ;;GET ADDRESS OF 1ST CHARACTER
037052  010037  037156                  MOV     R0,5$               ;;AND SAVE IT
037056  005001                          CLR     R1                  ;;CLEAR DATA WORD
037060  005002                          CLR     R2
037062  112046                  2$:     MOVB    (R0)+,-(SP)         ;;PICKUP THIS CHARACTER
037064  001420                          BEQ     3$                  ;;IF ZERO GET OUT
037066  122716  000060                  CMPB    #'0,(SP)            ;;MAKE SURE THIS CHARACTER
037072  003026                          BGT     4$                  ;;IS AN OCTAL DIGIT
037074  122716  000067                  CMPB    #'7,(SP)
037100  002423                          BLT     4$
037102  006301                          ASL     R1                  ;;*2
037104  006102                          ROL     R2
037106  006301                          ASL     R1                  ;;*4
037110  006102                          ROL     R2
037112  006301                          ASL     R1                  ;;*8
037114  006102                          ROL     R2
037116  042716  177770                  BIC     #↑C7,(SP)           ;;STRIP THE ASCII JUNK
037122  062601                          ADD     (SP)+,R1            ;;ADD IN THIS DIGIT
037124  000756                          BR      2$                  ;;LOOP
037126  005726                  3$:     TST     (SP)+               ;;CLEAN TERMINATOR FROM STACK
037130  010166  000012                  MOV     R1,12(SP)           ;;SAVE THE RESULT
037134  010237  037166                  MOV     R2,$HIOCT
037140  012602                          MOV     (SP)+,R2            ;;POP STACK INTO R2
037142  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
037144  012600                          MOV     (SP)+,R0            ;;POP STACK INTO R0
037146  000002                          RTI                         ;;RETURN
037150  005726                  4$:     TST     (SP)+               ;;CLEAN PARTIAL FROM STACK
037152  105010                          CLRB    (R0)                ;;SET A TERMINATOR
037154  104401                          TYPE                        ;;TYPE UP THRU THE BAD CHAR.
037156  000000                  5$:     .WORD   0
037160  104401  001244                  TYPE    ,$QUES              ;;"?" "CR" & "LF"
037164  000730                          BR      1$                  ;;TRY AGAIN
037166  000000                  $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

 037170                                 $SAVREG:
 037170  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 037172  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 037174  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 037176  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 037200  010446                                 MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 037202  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 037204  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
 037210  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
 037214  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
 037220  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
 037224  000002                                 RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
 037226                                 $RESREG:
 037226  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
 037232  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
 037236  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
 037242  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
 037246  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
 037250  012604                                 MOV     (SP)+,R4        ;;POP STACK INTO R4
 037252  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
 037254  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
 037256  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
 037260  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
 037262  000002                                 RTI

                                        .SBTTL  POWER DOWN AND UP ROUTINE

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

                                        ;POWER DOWN ROUTINE
 037264  017737  141650  002356         $PWRDN: MOV     @SWR,SAVSWR     ;SAVE SWITCH REGISTER
 037272  012737  037312  000024                 MOV     #$PWRUP,PWRVEC  ;SET UP VECTOR
 037300  012737  000340  000026                 MOV     #PR7,PWRVEC+2
 037306  000000                                 HALT
 037310  000776                                 BR      .-2     ;HANG UP

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

                                        ;POWER UP ROUTINE
 037312  005037  037406                 $PWRUP: CLR     $PWRCT          ;LOAD WAIT COUNT
 037316  012737  000144  037410                 MOV     #100.,$PWRCT+2
 037324  005237  037406                 1$:     INC     $PWRCT          ;WAIT FOR TELETYPE
 037330  001375                                 BNE     1$
 037332  005337  037410                         DEC     $PWRCT+2
```

```
037336  001372                          BNE     1$
037340  012737  037264  000024          MOV     #$PWRDN,PWRVEC  ;SET UP FOR POWER DOWN VECTOR
037346  012737  000340  000026          MOV     #PR7,PWRVEC+2
037354  012706  001100                  MOV     #STACK,SP       ;FORCE STACK
037360  104401  037412                  TYPE    ,$POWER         ;TYPE POWER
037364  004737  032516                  JSR     PC,PARCHK       ;REINITIALIZE MEMORY CHECK ENABLE
037370  013777  002356  141542          MOV     SAVSWR,@SWR     ;RESTORE SWITCH REGISTER
037376  013702  001324                  MOV     $BASE,R2        ;REINITIALISE R2 FOR '611 BASE
037402  000177  141500                  JMP     @$LPADR ;GO BACK TO LAST TEST

037406  000000  000000          $PWRCT: .WORD   0,0             ;TELETYPE TIME OUT
037412  047520  042527  000122  $POWER: .ASCIZ  /POWER/
                                        .EVEN
                                .SBTTL  TRAP DECODER

                                ;;*************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

037420  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
037422  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
037426  005740                          TST     -(R0)           ;;BACKUP BY 2
037430  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
037432  006300                          ASL     R0              ;;POSITION FOR INDEXING
037434  016000  037454                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
037440  000200                          RTS     R0              ;;GO TO ROUTINE



                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

037442  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
037444  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
037452  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                        -------
037454  037442                  $TRPAD: .WORD   $TRAP2
037456  034636                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
037460  035144                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
037462  035120                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
037464  035160                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
037466  035346                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

037470  036160                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

037472  036070                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
037474  036432                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
037476  036522                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
037500  037030                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
```

```
7859 037502 037170          $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
7860 037504 037226          $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
7861 037506 033566          SCOP1$  ;;CALL=SCOP1    TRAP+15(104415) INTERNAL LOOP ON ERROR
```

```
                                        .SBTTL  DATA TABLE FOR PRINT OUT

037510  001254  002310                  DT000:   .WORD   $TESTN,TRAPPC
037514  001254  001116  002220          DT0004:  .WORD   $TESTN,$ERRPC,E.CS1,T.CS1,T.CS2,T.ER,T.DS
037522  002160  002170  002174
037530  002172
037532  001254  001116  002244          DT0005:  .WORD   $TESTN,$ERRPC,E.MR1,T.MR1
037540  002204
037542  001254  001116  002246          DT0006:  .WORD   $TESTN,$ERRPC,E.MR2,T.MR2
037550  002206
037552  001254  001116  002250          DT0007:  .WORD   $TESTN,$ERRPC,E.MR3,T.MR3
037560  002210
037562  001254  001116  002244          DT0010:  .WORD   $TESTN,$ERRPC,E.MR1,T.MR1,PR.BIT,M1.BIT,BITCNT
037570  002204  002320  002322
037576  002326
037600  001254  001116  002220          DT0011:  .WORD   $TESTN,$ERRPC,E.CS1,T.CS1,PR.BIT,M1.BIT,BITCNT,T.CS2,T.ER,T.DS
037606  002160  002320  002322
037614  002326  002170  002174
037622  002172
037624  001254  001116  002240          DT0014:  .WORD   $TESTN,$ERRPC,E.DCYL,T.DCYL,E.DA,T.DA
037632  002200  002226  002166
037640  001254  001116  002244          DT0017:  .WORD   $TESTN,$ERRPC,E.MR1,T.MR1,P1.BIT,PR.BIT,M1.BIT,M2.BIT,BITCNT
037646  002204  002316  002320
037654  002322  002324  002326
037662  001254  001116  002254          DT0021:  .WORD   $TESTN,$ERRPC,E.ECPT,T.ECPT,BITCNT
037670  002214  002326
037674  001254  001116  002160          DT0026:  .WORD   $TESTN,$ERRPC,T.CS1,T.CS2,T.ER,T.DS,T.WC,T.DCYL,T.DA
037702  002170  002174  002172
037710  002162  002200  002166
037716  001254  001116                  DT0037:  .WORD   $TESTN,$ERRPC
037722  001162  001164  001166          DT0040:  .WORD   $REG0,$REG1,$REG2
037730  001254  001116  002254          DT0041:  .WORD   $TESTN,$ERRPC,E.ECPT,T.ECPT,E.ECPS,T.ECPS,BITCNT
037736  002214  002252  002212
037744  002326
037746  001254  001116  002252          DT0043:  .WORD   $TESTN,$ERRPC,E.ECPS,T.ECPS,BITCNT
037754  002212  002326
037760  001254  001116  002234          DT0045:  .WORD   $TESTN,$ERRPC,E.ER,T.ER
037766  002174
037770  001254  001116  002224          DT0046:  .WORD   $TESTN,$ERRPC,E.BA,T.BA
037776  002164
040000  001254  001116  002222          DT0062:  .WORD   $TESTN,$ERRPC,E.WC,T.WC
040006  002162
040010  001254  001116  001162          DT0063:  .WORD   $TESTN,$ERRPC,$REG0,$REG3,$REG1,$REG4,$REG2
040016  001170  001164  001172
040024  001166
```

```
                                  .SBTTL  DATA FORMAT FOR PRINT OUT

040026  000001            DF000:  .WORD   1
040030     002     000            .BYTE   2,0
040032  000006            DF0004: .WORD   6
040034     000     000            .BYTE   0,0
040036  041232                    .WORD   DH000A
040040     000     000            .BYTE   0,0
040042  041250                    .WORD   DH000B
040044     002     000            .BYTE   2,0
040046  041314                    .WORD   DH004
040050     000     000            .BYTE   0,0
040052  041333                    .WORD   DH004A
040054     002     000            .BYTE   2,0
040056  041423                    .WORD   DH004E
040060     003     000            .BYTE   3,0
040062  000005            DF0005: .WORD   5
040064     000     000            .BYTE   0,0
040066  041232                    .WORD   DH000A
040070     000     000            .BYTE   0,0
040072  041250                    .WORD   DH000B
040074     002     000            .BYTE   2,0
040076  041314                    .WORD   DH004
040100     000     000            .BYTE   0,0
040102  041351                    .WORD   DH004B
040104     002     000            .BYTE   2,0
040106  000005            DF0006: .WORD   5
040110     000     000            .BYTE   0,0
040112  041232                    .WORD   DH000A
040114     000     000            .BYTE   0,0
040116  041250                    .WORD   DH000B
040120     002     000            .BYTE   2,0
040122  041314                    .WORD   DH004
040124     000     000            .BYTE   0,0
040126  041367                    .WORD   DH004C
040130     002     000            .BYTE   2,0
040132  000005            DF0007: .WORD   5
040134     000     000            .BYTE   0,0
040136  041232                    .WORD   DH000A
040140     000     000            .BYTE   0,0
040142  041250                    .WORD   DH000B
040144     002     000            .BYTE   2,0
040146  041314                    .WORD   DH004
040150     000     000            .BYTE   0,0
040152  041405                    .WORD   DH004D
040154     002     000            .BYTE   2,0
040156  000005            DF0010: .WORD   5
040160     000     000            .BYTE   0,0
040162  041232                    .WORD   DH000A
040164     000     000            .BYTE   0,0
040166  041250                    .WORD   DH000B
040170     002     000            .BYTE   2,0
040172  041517                    .WORD   DH010
040174     000     000            .BYTE   0,0
040176  041563                    .WORD   DH010A
040200     005     000            .BYTE   5,0
```

```
040202  000006                 DF0011: .WORD   6
040204     000        000              .BYTE   0,0
040206  041232                         .WORD   DH000A
040210     000        000              .BYTE   0,0
040212  041250                         .WORD   DH000B
040214     002        000              .BYTE   2,0
040216  041517                         .WORD   DH010
040220     000        000              .BYTE   0,0
040222  041631                         .WORD   DH010B
040224     005        000              .BYTE   5,0
040226  041423                         .WORD   DH004E
040230     003        000              .BYTE   3,0
040232  000005                 DF0014: .WORD   5
040234     000        000              .BYTE   0,0
040236  041232                         .WORD   DH000A
040240     000        000              .BYTE   0,0
040242  041250                         .WORD   DH000B
040244     002        000              .BYTE   2,0
040246  041677                         .WORD   DH014
040250     000        000              .BYTE   0,0
040252  041736                         .WORD   DH014A
040254     004        000              .BYTE   4,0
040256  000005                 DF0017: .WORD   5
040260     000        000              .BYTE   0,0
040262  041232                         .WORD   DH000A
040264     000        000              .BYTE   0,0
040266  041250                         .WORD   DH000B
040270     002        000              .BYTE   2,0
040272  041773                         .WORD   DH017
040274     000        000              .BYTE   0,0
040276  042057                         .WORD   DH017A
040300     007        000              .BYTE   7,0
040302  000005                 DF0021: .WORD   5
040304     000        000              .BYTE   0,0
040306  041232                         .WORD   DH000A
040310     000        000              .BYTE   0,0
040312  041250                         .WORD   DH000B
040314     002        000              .BYTE   2,0
040316  042145                         .WORD   DH021
040320     000        000              .BYTE   0,0
040322  042171                         .WORD   DH021A
040324     003        000              .BYTE   3,0
040326  000004                 DF0026: .WORD   4
040330     000        000              .BYTE   0,0
040332  041232                         .WORD   DH000A
040334     000        000              .BYTE   0,0
040336  041250                         .WORD   DH000B
040340     002        000              .BYTE   2,0
040342  042245                         .WORD   DH026
040344     007        000              .BYTE   7,0
040346  000003                 DF0027: .WORD   3
040350     000        000              .BYTE   0,0
040352  041250                         .WORD   DH000B
040354     002        000              .BYTE   2,0
040356  042245                         .WORD   DH026
040360     007        000              .BYTE   7,0
```

```
8019  040362  000004            DF0037: .WORD   4
8020  040364     000     000            .BYTE   0,0
8021  040366  041250                    .WORD   DH000B
8022  040370     002     000            .BYTE   2,0
8023  040372  042332                    .WORD   DH037A
8024  040374     000     000            .BYTE   0,0
8025  040376  042357                    .WORD   DH037B
8026  040400     003     000            .BYTE   3,0
8027  040402  000000            DF0040: .WORD   0
8028  040404     003     000            .BYTE   3,0
8029  040406  000004            DF0041: .WORD   4
8030  040410     000     000            .BYTE   0,0
8031  040412  041250                    .WORD   DH000B
8032  040414     002     000            .BYTE   2,0
8033  040416  042406                    .WORD   DH0041
8034  040420     000     000            .BYTE   0,0
8035  040422  042452                    .WORD   DH041A
8036  040424     005     000            .BYTE   5,0
8037  040426  000004            DF0042: .WORD   4
8038  040430     000     000            .BYTE   0,0
8039  040432  041250                    .WORD   DH000B
8040  040434     002     000            .BYTE   2,0
8041  040436  042145                    .WORD   DH0021
8042  040440     000     000            .BYTE   0,0
8043  040442  042171                    .WORD   DH021A
8044  040444     003     000            .BYTE   3,0
8045  040446  000004            DF0043: .WORD   4
8046  040450     000     000            .BYTE   0,0
8047  040452  041250                    .WORD   DH000B
8048  040454     002     000            .BYTE   2,0
8049  040456  042145                    .WORD   DH0021
8050  040460     000     000            .BYTE   0,0
8051  040462  042217                    .WORD   DH021B
8052  040464     003     000            .BYTE   3,0
8053  040466  000005            DF0045: .WORD   5
8054  040470     000     000            .BYTE   0,0
8055  040472  041232                    .WORD   DH000A
8056  040474     000     000            .BYTE   0,0
8057  040476  041250                    .WORD   DH000B
8058  040500     002     000            .BYTE   2,0
8059  040502  041314                    .WORD   DH004
8060  040504     000     000            .BYTE   0,0
8061  040506  041450                    .WORD   DH004F
8062  040510     002     000            .BYTE   2,0
8063  040512  000004            DF0046: .WORD   4
8064  040514     000     000            .BYTE   0,0
8065  040516  041250                    .WORD   DH000B
8066  040520     002     000            .BYTE   2,0
8067  040522  041314                    .WORD   DH004
8068  040524     000     000            .BYTE   0,0
8069  040526  041465                    .WORD   DH004G
8070  040530     002     000            .BYTE   2,0
8071  040532  000005            DF0050: .WORD   5
8072  040534     000     000            .BYTE   0,0
8073  040536  041232                    DH000A
8074  040540     000     000            .BYTE   0,0
```

```
040542  041250                      .WORD   DH000B
040544     002     000              .BYTE   2,0
040546  041314                      .WORD   DH004
040550     000     000              .BYTE   0,0
040552  041333                      .WORD   DH004A
040554     002     000              .BYTE   2,0
040556  000004            DF0060:   .WORD   4
040560     000     000              .BYTE   0,0
040562  041250                      .WORD   DH000B
040564     002     000              .BYTE   2,0
040566  042520                      .WORD   DH060A
040570     000     000              .BYTE   0,0
040572  042545                      .WORD   DH060B
040574     003     000              .BYTE   3,0
040576  000005            DF0061:   .WORD   5
040600     000     000              .BYTE   0,0
040602  041232                      .WORD   DH000A
040604     000     000              .BYTE   0,0
040606  041250                      .WORD   DH000B
040610     002     000              .BYTE   2,0
040612  041314                      .WORD   DH004
040614     000     000              .BYTE   0,0
040616  041465                      .WORD   DH004G
040620     002     000              .BYTE   2,0
040622  000005            DF0062:   .WORD   5
040624     000     000              .BYTE   0,0
040626  041232                      .WORD   DH000A
040630     000     000              .BYTE   0,0
040632  041250                      .WORD   DH000B
040634     002     000              .BYTE   2,0
040636  041314                      .WORD   DH004
040640     000     000              .BYTE   0,0
040642  041502                      .WORD   DH004H
040644     002     000              .BYTE   2,0
040646  000005            DF0063:   .WORD   5
040650     000     000              .BYTE   0,0
040652  041232                      .WORD   DH000A
040654     000     000              .BYTE   0,0
040656  041250                      .WORD   DH000B
040660     002     000              .BYTE   2,0
040662  042572                      .WORD   DH063A
040664     000     000              .BYTE   0,0
C40666  042637                      .WORD   DH063B
040670     005     000              .BYTE   5,0
```

J12


```
                                                .SBTTL  ASCII MESSAGES

040672  005015  045522  030466  OPR001: .ASCIZ  <15><12>/RK611 VECTOR ADDRESS ( /
040700  020061  042526  052103
040706  051117  040440  042104
040714  042522  051523  024040
040722  000040
040724  024440  036440  000040  OPR002: .ASCIZ  / ) = /
040732  045522  030466  020061  OPR003: .ASCIZ  /RK611 VECTOR ADDRESS ( /
040740  042526  052103  051117
040746  040440  042104  042522
040754  051523  024040  000040
040762  045522  030466  020061  OPR004: .ASCIZ  /RK611 PRIORITY ( /
040770  051120  047511  044522
040776  054524  024040  000040
041004  005015  047062  020104  OPR006: .ASCIZ  <15><12>/2ND PASS RUN TIME APPROX 4 MINUTES/<15><12>
041012  040520  051523  051040
041020  047125  052040  046511
041026  020105  050101  051120
041034  054117  032040  046440
041042  047111  052125  051505
041050  005015     000
041053     015  025012  025052  OPR007: .ASCIZ  <15><12>/*****    PROGRAM HALTED    *****/<15><12>
041060  025052  020040  050040
041066  047522  051107  046501
041074  044040  046101  042524
041102  020104  020040  025052
041110  025052  006452  000012
041116  020040     000          SPACE2: .ASCIZ  /  /
041121     015  050012  047522  ABORT:  .ASCIZ  <15><12>/PROGRAM ABORTED BECAUSE ERROR THRESHOLD EXCEEDED/<15><12>
041126  051107  046501  040440
041134  047502  052122  042105
041142  041040  041505  052501
041150  042523  042440  051122
041156  051117  052040  051110
041164  051505  047510  042114
041172  042440  041530  042505
041200  042504  006504  000012
041206  005015  042524  052123  TSTBY1: .ASCIZ  <15><12>/TEST /
041214  000040
041216  041040  050131  051501  TSTBY2: .ASCIZ  / BYPASSED/<15><12>
041224  042523  006504  000012
```

```
                                                .SBTTL  DATA HEADERS

041232  042524  052123  020040  DH000A: .ASCIZ  /TEST    ERROR/
041240  020040  051105  047522
041246  000122
041250  052516  020115  020040  DH000B: .ASCIZ  /NUM     PC/
041256  020040  041520     000
041263     124  051505  020124  DH000C: .ASCII  /TEST    TRAP/<15><12>
041270  020040  052040  040522
041276  006520     012
041301     116  046525  020040          .ASCIZ  /NUM     PC/
041306  020040  050040  000103
041314  054105  042520  052103  DH004:  .ASCIZ  /EXPECT  ACTUAL/
041322  020040  041501  052524
041330  046101     000
041333     122  041513  030523  DH004A: .ASCIZ  /RKCS1   RKCS1/
041340  020040  051040  041513
041346  030523     000
041351     122  046513  030522  DH004B: .ASCIZ  /RKMR1   RKMR1/
041356  020040  051040  046513
041364  030522     000
041367     122  046513  031122  DH004C: .ASCIZ  /RKMR2   RKMR2/
041374  020040  051040  046513
041402  031122     000
041405     122  046513  031522  DH004D: .ASCIZ  /RKMR3   RKMR3/
041412  020040  051040  046513
041420  031522     000
041423     122  041513  031123  DH004E: .ASCIZ  /RKCS2   RKER    RKDS/
041430  020040  051040  042513
041436  020122  020040  051040
041444  042113  000123
041450  045522  051105  020040  DH004F: .ASCIZ  /RKER    RKER/
041456  020040  045522  051105
041464     000
041465     122  041113  020101  DH004G: .ASCIZ  /RKBA    RKBA/
041472  020040  051040  041113
041500  000101
041502  045522  041527  020040  DH004H: .ASCIZ  /RKWC    RKWC/
041510  020040  045522  041527
041516     000
041517     105  050130  041505  DH010:  .ASCIZ  /EXPECT  ACTUAL  PRESENT PREVOUS BIT/
041524  020124  040440  052103
041532  040525  020114  050040
041540  042522  042523  052116
041546  050040  042522  047526
041554  051525  041040  052111
041562     000
041563     122  046513  030522  DH010A: .ASCIZ  /RKMR1   RKMR1   BIT     BIT     COUNT/
041570  020040  051040  046513
041576  030522  020040  041040
041604  052111  020040  020040
041612  041040  052111  020040
041620  020040  041440  052517
041626  052116     000
041631     122  041513  030523  DH010B: .ASCIZ  /RKCS1   RKCS1   BIT     BIT     COUNT/
041636  020040  051040  041513
```

```
041644  030523  020040  041040
041652  052111  020040  020040
041660  041040  052111  020040
041666  020040  041440  052517
041674  052116     000
041677     105  050130  041505  DH014:  .ASCIZ  /EXPECT  ACTUAL  EXPECT  ACTUAL/
041704  020124  040440  052103
041712  040525  020114  042440
041720  050130  041505  020124
041726  040440  052103  040525
041734  000114
041736  045522  041504  046131  DH014A: .ASCIZ  /RKDCYL  RKDCYL  RKDA    RKDA/
041744  020040  045522  041504
041752  046131  020040  045522
041760  040504  020040  020040
041766  045522  040504     000
041773     105  050130  041505  DH0017: .ASCIZ  /EXPECT  ACTUAL  PRESENT PRESENT PRESENT PRESENT BIT/
042000  020124  040440  052103
042006  040525  020114  050040
042014  042522  042523  052116
042022  050040  042522  042523
042030  052116  050040  042522
042036  042523  052116  050040
042044  042522  042523  052116
042052  041040  052111     000
042057     122  046513  030522  DH017A: .ASCIZ  /RKMR1   RKMR1   BIT+1   BIT     BIT-1   BIT-2   COUNT/
042064  020040  051040  046513
042072  030522  020040  041040
042100  052111  030453  020040
042106  041040  052111  020040
042114  020040  041040  052111
042122  030455  020040  041040
042130  052111  031055  020040
042136  041440  052517  052116
042144     000
042145     105  050130  041505  DH0021: .ASCIZ  /EXPECT  ACTUAL  BIT/
042152  020124  040440  052103
042160  040525  020114  041040
042166  052111     000
042171     122  042513  050103  DH021A: .ASCIZ  /RKECPT  RKECPT  COUNT/
042176  020124  051040  042513
042204  050103  020124  041440
042212  052517  052116     000
042217     122  042513  050103  DH021B: .ASCIZ  /RKECPS  RKECPS  COUNT/
042224  020123  051040  042513
042232  050103  020123  041440
042240  052517  052116     000
042245     122  041513  030523  DH0026: .ASCIZ  /RKCS1   RKCS2   RKER    RKDS    RKWC    RKDCYL  RKDA/
042252  020040  051040  041513
042260  031123  020040  051040
042266  042513  020122  020040
042274  051040  042113  020123
042302  020040  051040  053513
042310  020103  020040  051040
042316  042113  054503  020114
042324  051040  042113  000101
```

```
042332  047507  042117  020040  DH037A: .ASCIZ  /GOOD    BAD     WORD/
042340  020040  040502  020104
042346  020040  020040  047527
042354  042122     000
042357     127  051117  020104  DH037B: .ASCIZ  /WORD    WORD    NUMBER/
042364  020040  053440  051117
042372  020104  020040  047040
042400  046525  042502  000122
042406  054105  042520  052103  DH0041: .ASCIZ  /EXPECT  ACTUAL  EXPECT  ACTUAL  BIT/
042414  020040  041501  052524
042422  046101  020040  054105
042430  042520  052103  020040
042436  041501  052524  046101
042444  020040  044502  000124
042452  045522  041505  052120  DH041A: .ASCIZ  /RKECPT  RKECPT  RKECPS  RKECPS  COUNT/
042460  020040  045522  041505
042466  052120  020040  045522
042474  041505  051520  020040
042502  045522  041505  051520
042510  020040  047503  047125
042516  000124
042520  045522  051503  020062  DH060A: .ASCIZ  /RKCS2   BUFFER  WORD/
042526  020040  052502  043106
042534  051105  020040  047527
042542  042122     000
042545     040  020040  020040  DH060B: .ASCIZ  /        WORD    READ/
042552  020040  053440  051117
042560  020104  020040  051040
042566  040505  000104
042572  045522  051503  020062  DH063A: .ASCIZ  /RKCS2   BA17-16 BUFFER  BA17-16 WORD/
042600  020040  040502  033461
042606  030455  020066  052502
042614  043106  051105  020040
042622  040502  033461  030455
042630  020066  047527  042122
042636     000
042637     040  020040  020040  DH063B: .ASCIZ  /        BUF WD  WORD    WD READ READ/
042644  020040  041040  043125
042652  053440  020104  053440
042660  051117  020104  020040
042666  053440  020104  042522
042674  042101  051040  040505
042702  000104
```

```
                                        .SBTTL  ERROR MESSAGES

 042704  047125  054105  042520  EM000:  .ASCIZ  /UNEXPECTED MEMORY PARITY ENABLE TRAP/
 042712  052103  042105  046440
 042720  046505  051117  020131
 042726  040520  044522  054524
 042734  042440  040516  046102
 042742  020105  051124  050101
 042750     000
 042751     105  051122  051117  EM5004: .ASCIZ  /ERROR AFTER ATTEMPTING IMPLIED SEEK/
 042756  040440  052106  051105
 042764  040440  052124  046505
 042772  052120  047111  020107
 043000  046511  046120  042511
 043006  020104  042523  045505
 043014     000
 043015     122  041513  030523  E5004A: .ASCIZ  /RKCS1 IN ERROR/
 043022  044440  020116  051105
 043030  047522  000122
 043034  045522  051115  020061  E5004B: .ASCIZ  /RKMR1 IN ERROR/
 043042  047111  042440  051122
 043050  051117     000
 043053     122  046513  031122  E5004C: .ASCIZ  /RKMR2 IN ERROR/
 043060  044440  020116  051105
 043066  047522  000122
 043072  045522  051115  020063  E5004D: .ASCIZ  /RKMR3 IN ERROR/
 043100  047111  042440  051122
 043106  051117     000
 043111     122  042513  020122  E5004E: .ASCIZ  /RKER IN ERROR/
 043116  047111  042440  051122
 043124  051117     000
 043127     122  041113  020101  E5004F: .ASCIZ  /RKBA IN ERROR/
 043134  047111  042440  051122
 043142  051117     000
 043145     122  053513  020103  E5004G: .ASCIZ  /RKWC IN ERROR/
 043152  047111  042440  051122
 043160  051117     000
 043163     105  051122  051117  EM5010: .ASCIZ  /ERROR ATTEMPTING TO READ HEADER SYNC (PREAMBLE)/
 043170  040440  052124  046505
 043176  052120  047111  020107
 043204  047524  051040  040505
 043212  020104  042510  042101
 043220  051105  051440  047131
 043226  020103  050050  042522
 043234  046501  046102  024505
 043242     000
 043243     105  051122  051117  EM5011: .ASCIZ  &ERROR ATTEMPTING TO DO HEADER READ/COMPARE&
 043250  040440  052124  046505
 043256  052120  047111  020107
 043264  047524  042040  020117
 043272  042510  042101  051105
 043300  051040  040505  027504
 043306  047503  050115  051101
 043314  000105
 043316  045522  041504  046131  E5011A: .ASCIZ  /RKDCYL OR RKDA IN ERROR/
 043324  047440  020122  045522
```

```
043332  040504  044440  020116
043340  051105  047522  000122
043346  051105  047522  020122  EM5015: .ASCIZ  /ERROR ATTEMPTING TO READ GAP/
043354  052101  042524  050115
043362  044524  043516  052040
043370  020117  042522  042101
043376  043440  050101     000
043403     105  051122  051117  EM5020: .ASCIZ  /ERROR ATTEMPTING TO WRITE DATA SYNC (PREAMBLE)/
043410  040440  052124  046505
043416  052120  047111  020107
043424  047524  053440  044522
043432  042524  042040  052101
043440  020101  054523  041516
043446  024040  051120  040505
043454  041115  042514  000051
043462  051105  047522  020122  EM5021: .ASCIZ  /ERROR ATTEMPTING TO WRITE DATA OR ECC/
043470  052101  042524  050115
043476  044524  043516  052040
043504  020117  051127  052111
043512  020105  040504  040524
043520  047440  020122  041505
043526  000103
043530  051105  047522  020122  E5021A: .ASCIZ  /ERROR IN ECC GENERATION/
043536  047111  042440  041503
043544  043440  047105  051105
043552  052101  047511  000116
043560  051105  047522  020122  EM5023: .ASCIZ  /ERROR BETWEEN SECTORS ATTEMPTING MULTI-SECTOR WRITE/
043566  042502  053524  042505
043574  020116  042523  052103
043602  051117  020123  052101
043610  042524  050115  044524
043616  043516  046440  046125
043624  044524  051455  041505
043632  047524  020122  051127
043640  052111  000105
043644  051115  020061  047111  EMW1:   .ASCIZ  /MR1 INCORRECT ON 1ST UPWARD TRANSITION OF MAINT CLOCK/
043652  047503  051122  041505
043660  020124  047117  030440
043666  052123  052440  053520
043674  051101  020104  051124
043702  047101  044523  044524
043710  047117  047440  020106
043716  040515  047111  020124
043724  046103  041517  000113
043732  051115  020061  047111  EMW2:   .ASCIZ  /MR1 INCORRECT ON 1ST DOWNWARD TRANSITION OF MAINT CLOCK/
043740  047503  051122  041505
043746  020124  047117  030440
043754  052123  042040  053517
043762  053516  051101  020104
043770  051124  047101  044523
043776  044524  047117  047440
044004  020106  040515  047111
044012  020124  046103  041517
044020  000113
044022  051115  020061  047111  EMW3:   .ASCIZ  /MR1 INCORRECT ON 2ND UPWARD TRANSITION OF MAINT CLOCK/
044030  047503  051122  041505
```

```
 8428  044036  020124  047117  031040
 8429  044044  042116  052440  053520
 8430  044052  051101  020104  051124
 8431  044060  047101  044523  044524
 8432  044066  047117  047440  020106
 8433  044074  040515  047111  020124
 8434  044102  046103  041517  000113
 8435  044110  051115  020061  047111  EMW4:   .ASCIZ  /MR1 INCORRECT ON 2ND DOWNWARD TRANSITION OF MAINT CLOCK/
 8436  044116  047503  051122  041505
 8437  044124  020124  047117  031040
 8438  044132  042116  042040  053517
 8439  044140  053516  051101  020104
 8440  044146  051124  047101  044523
 8441  044154  044524  047117  047440
 8442  044162  020106  040515  047111
 8443  044170  020124  046103  041517
 8444  044176  000113
 8445  044200  054105  042520  052103  EM5026: .ASCIZ  /EXPECTED CYLINDER OVERFLOW ERR DID NOT SET/
 8446  044206  042105  041440  046131
 8447  044214  047111  042504  020122
 8448  044222  053117  051105  046106
 8449  044230  053517  042440  051122
 8450  044236  042040  042111  047040
 8451  044244  052117  051440  052105
 8452  044252     000
 8453  044253     127  044522  042524  E5026A: .ASCIZ  /WRITE TO CYL 632, TRACK 2, SECTOR 25, WORD COUNT 401/
 8454  044260  052040  020117  054503
 8455  044266  020114  031466  026062
 8456  044274  052040  040522  045503
 8457  044302  031040  020054  042523
 8458  044310  052103  051117  031040
 8459  044316  026065  053440  051117
 8460  044324  020104  047503  047125
 8461  044332  020124  030064  000061
 8462  044340  047125  054105  042520  EM5027: .ASCIZ  /UNEXPECTED ERROR WHILE TRYING TO FORCE CYL OVERFLOW/
 8463  044346  052103  042105  042440
 8464  044354  051122  051117  053440
 8465  044362  044510  042514  052040
 8466  044370  054522  047111  020107
 8467  044376  047524  043040  051117
 8468  044404  042503  041440  046131
 8469  044412  047440  042526  043122
 8470  044420  047514  000127
 8471  044424  051127  052111  020105  E5026B: .ASCIZ  /WRITE TO CYL 1456, TRACK 2, SECTOR 25, WORD COUNT 401/
 8472  044432  047524  041440  046131
 8473  044440  030440  032464  026066
 8474  044446  052040  040522  045503
 8475  044454  031040  020054  042523
 8476  044462  052103  051117  031040
 8477  044470  026065  053440  051117
 8478  044476  020104  047503  047125
 8479  044504  020124  030064  000061
 8480  044512  047125  054105  042520  EM5030: .ASCIZ  /UNEXPECTED CYLINDER OVERFLOW WRITING LAST PHYSICAL ADDRESS/
 8481  044520  052103  042105  041440
 8482  044526  046131  047111  042504
 8483  044534  020122  053117  051105
```

```
044542  046106  053517  053440
044550  044522  044524  043516
044556  046040  051501  020124
044564  044120  051531  041511
044572  046101  040440  042104
044600  042522  051523     000
044605     125  042516  050130  EM5031: .ASCIZ  /UNEXPECTED ERROR WHILE WRITING LAST PHYSICAL ADDRESS/
044612  041505  042524  020104
044620  051105  047522  020122
044626  044127  046111  020105
044634  051127  052111  047111
044642  020107  040514  052123
044650  050040  054510  044523
044656  040503  020114  042101
044664  051104  051505  000123
044672  051105  047522  020122  EM5032: .ASCIZ  /ERROR ATTEMPTING TO READ DATA SYNC (PREAMBLE)/
044700  052101  042524  050115
044706  044524  043516  052040
044714  020117  042522  042101
044722  042040  052101  020101
044730  054523  041516  024040
044736  051120  040505  041115
044744  042514  000051
044750  051105  047522  020122  EM5034: .ASCIZ  /ERROR ATTEMPTING TO READ DATA OR ECC/
044756  052101  042524  050115
044764  044524  043516  052040
044772  020117  042522  042101
045000  042040  052101  020101
045006  051117  042440  041503
045014     000
045015     105  041503  042440  EM5035: .ASCIZ  /ECC ERROR IN READ DATA OPERATION/
045022  051122  051117  044440
045030  020116  042522  042101
045036  042040  052101  020101
045044  050117  051105  052101
045052  047511  000116
045056  040504  040524  041440  EM5037: .ASCIZ  /DATA COMPARE ERROR AFTER INPUT NPR TRANSFER/
045064  046517  040520  042522
045072  042440  051122  051117
045100  040440  052106  051105
045106  044440  050116  052125
045114  047040  051120  052040
045122  040522  051516  042506
045130  000122
045132  041505  020103  042522  EM5041: .ASCIZ  /ECC REGISTER NOT CORRECT AFTER ECC READ IN READ DATA/
045140  044507  052123  051105
045146  047040  052117  041440
045154  051117  042522  052103
045162  040440  052106  051105
045170  042440  041503  051040
045176  040505  020104  047111
045204  051040  040505  020104
045212  040504  040524     000
045217     105  051122  051117  EM5042: .ASCIZ  /ERROR IN ECC PATTERN CALCULATION AFTER ECC ERROR/
045224  044440  020116  041505
045232  020103  040520  052124
```

```
045240  051105  020116  040503
045246  041514  046125  052101
045254  047511  020116  043101
045262  042524  020122  041505
045270  020103  051105  047522
045276  000122
045300  051105  047522  020122   EM5043: .ASCIZ  /ERROR IN ECC POSITION COUNTING AFTER ECC ERROR/
045306  047111  042440  041503
045314  050040  051517  052111
045322  047511  020116  047503
045330  047125  044524  043516
045336  040440  052106  051105
045344  042440  041503  042440
045352  051122  051117     000
045357     105  051122  051117   EM5044: .ASCIZ  /ERROR AFTER PROCESSING DATA CHECK ERROR/
045364  040440  052106  051105
045372  050040  047522  042503
045400  051523  047111  020107
045406  040504  040524  041440
045414  042510  045503  042440
045422  051122  051117     000
045427     124  040522  051516   EM5046: .ASCIZ  /TRANSFER LENGTH ERROR ATTEMPTING PARTIAL SECTOR OPERATION/
045434  042506  020122  042514
045442  043516  044124  042440
045450  051122  051117  040440
045456  052124  046505  052120
045464  047111  020107  040520
045472  052122  040511  020114
045500  042523  052103  051117
045506  047440  042520  040522
045514  044524  047117     000
045521     102  042101  051440   EM5047: .ASCIZ  /BAD SECTOR ERROR DID NOT PREVENT DESIRED ADDRESS INCREMENT/
045526  041505  047524  020122
045534  051105  047522  020122
045542  044504  020104  047516
045550  020124  051120  053105
045556  047105  020124  042504
045564  044523  042522  020104
045572  042101  051104  051505
045600  020123  047111  051103
045606  046505  047105  000124
045614  040502  020104  042523   EM5050: .ASCIZ  /BAD SECTOR ERROR DID NOT CAUSE CERR TO SET/
045622  052103  051117  042440
045630  051122  051117  042040
045636  042111  047040  052117
045644  041440  052501  042523
045652  041440  051105  020122
045660  047524  051440  052105
045666     000
045667     102  042101  051440   EM5051: .ASCIZ  /BAD SECTOR ERROR FAILED TO SET WHEN EXPECTED/
045674  041505  047524  020122
045702  051105  047522  020122
045710  040506  046111  042105
045716  052040  020117  042523
045724  020124  044127  047105
045732  042440  050130  041505
```

```
045740  042524  000104
045744  040504  040524  052040  EM5052: .ASCII  /DATA TRANSFER DID NOT CONTINUE TO END OF SECTOR/
045752  040522  051516  042506
045760  020122  044504  020104
045766  047516  020124  047503
045774  052116  047111  042525
046002  052040  020117  047105
046010  020104  043117  051440
046016  041505  047524     122
046023     015  053412  042510          .ASCIZ  <15><12>/WHEN BAD SECTOR ERROR OCCURRED/
046030  020116  040502  020104
046036  042523  052103  051117
046044  042440  051122  051117
046052  047440  041503  051125
046060  042522  000104
046064  051105  047522  020122  EM5053: .ASCIZ  /ERROR ATTEMPTING WRITE CHECK OR ECC READ AFTER WRT CHK/
046072  052101  042524  050115
046100  044524  043516  053440
046106  044522  042524  041440
046114  042510  045503  047440
046122  020122  041505  020103
046130  042522  042101  040440
046136  052106  051105  053440
046144  052122  041440  045510
046152     000
046153     105  041503  042440  EM5054: .ASCIZ  /ECC ERROR IN WRITE CHECK OPERATION/
046160  051122  051117  044440
046166  020116  051127  052111
046174  020105  044103  041505
046202  020113  050117  051105
046210  052101  047511  000116
046216  051127  052111  020105  EM5060: .ASCIZ  /WRITE CHECK ERROR FAILED TO SET IN 16 BIT MODE/
046224  044103  041505  020113
046232  051105  047522  020122
046240  040506  046111  042105
046246  052040  020117  042523
046254  020124  047111  030440
046262  020066  044502  020124
046270  047515  042504     000
046275     124  040522  051516  EM5061: .ASCIZ  /TRANSFER LENGTH ERROR WITH WRITE CHECK ERROR/
046302  042506  020122  042514
046310  043516  044124  042440
046316  051122  051117  053440
046324  052111  020110  051127
046332  052111  020105  044103
046340  041505  020113  051105
046346  047522  000122
046352  051127  052111  020105  EM5063: .ASCIZ  /WRITE CHECK ERROR FAILED TO SET IN 18 BIT MODE/
046360  044103  041505  020113
046366  051105  047522  020122
046374  040506  046111  042105
046402  052040  020117  042523
046410  020124  047111  030440
046416  020070  044502  020124
046424  047515  042504     000
```

```
                046432                    .EVEN
                                  .SBTTL  DATA PATTERNS AND BUFFERS
                                  ;*      PATTERN 1 - ALL ZEROS
                                  ;*      PATTERN 2 - ROTATING CELL PULSE PRECOMP

        046432                    PAT2:
        046432  121105                    121105
        046434  150442                    150442
        046436  064221                    064221
        046440  132110                    132110
        046442  055044                    055044
        046444  026422                    026422
        046446  013211                    013211
        046450  105504                    105504
        046452  042642                    042642
        046454  021321                    021321
        046456  110550                    110550
        046460  044264                    044264
        046462  022132                    022132
        046464  011055                    011055
        046466  104426                    104426
        046470  042213                    042213

                                  ;*      PATTERN 3 - MAX PRECOMP PHASE MIX

        046472                    PAT3:
        046472  133333                    133333
        046474  066666                    066666
        046476  155555                    155555
        046500  155555                    155555
        046502  133333                    133333
        046504  066666                    066666
        046506  066666                    066666
        046510  155555                    155555
        046512  155555                    155555
        046514  133333                    133333
        046516  133333                    133333
        046520  133333                    133333
        046522  133333                    133333
        046524  133333                    133333
        046526  133333                    133333
        046530  133333                    133333

                                  ;*      PATTERN 4 - HI-LO FREQ. MIX

        046532                    PAT4:
        046532  177777                    177777
        046534  177777                    177777
        046536  177777                    177777
        046540  052525                    052525
        046542  052525                    052525
        046544  052525                    052525
        046546  177777                    177777
        046550  177777                    177777
        046552  052525                    052525
        046554  052525                    052525
```

```
      046556  177777                 177777
      046560  052525                 052525
      046562  177252                 177252
      046564  177252                 177252
      046566  172765                 172765
      046570  172765                 172765

                             ;*      PATTERN 5 - ALTERNATING 1'S AND 0'S

                             ;*      PATTERN 6 - COMPOSIT ROTATING

      046572                 PAT6:
      046572  072307                 072307
      046574  135143                 135143
      046576  156461                 156461
      046600  167230                 167230
      046602  073514                 073514
      046604  035646                 035646
      046606  016723                 016723
      046610  107351                 107351
      046612  143564                 143564
      046614  061672                 061672
      046616  030735                 030735
      046620  114356                 114356
      046622  046167                 046167
      046624  123073                 123073
      046626  151453                 151453
      046630  164616                 164616

                             ;*      PATTERN 7 - ALL ONES

      046632  000402         IBUFF:  .BLKW   402
      047636  000402         OBUFF:  .BLKW   402
              000001         .END
```

```
ABASE = 177440      1053#   1306    1347
ABORT   041121      7180    8148#
ACDW1 = 000000      1306
ACDW2 = 000000      1306
ACLO  = 000010      1148#
ACPUOP= 000000      1306    1321
ADDW0 = 000000      1306
ADDW1 = 000000      1306
ADDW10= 000000      1306
ADDW11= 000000      1306
ADDW12= 000000      1306
ADDW13= 000000      1306
ADDW14= 000000      1306
ADDW15= 000000      1306
ADDW2 = 000000      1306
ADDW3 = 000000      1306
ADDW4 = 000000      1306
ADDW5 = 000000      1306
ADDW6 = 000000      1306
ADDW7 = 000000      1306
ADDW8 = 000000      1306
ADDW9 = 000000      1306
ADEVCT= 000000      1306    1312
ADEVM = 000000      1306    1348
AENV  = 000000      1306    1317
AENVM = 000000      1306    1318
AFATAL= 000000      1306    1309
AMADR1= 000000      1306    1334
AMADR2= 000000      1306    1338
AMADR3= 000000      1306    1341
AMADR4= 000000      1306    1344
AMAMS1= 000000      1306    1328
AMAMS2= 000000      1306    1336
AMAMS3= 000000      1306    1339
AMAMS4= 000000      1306    1342
AMSGAD= 000000      1306    1314
AMSGLG= 000000      1306    1315
AMSGTY= 000000      1306    1308
AMTYP1= 000000      1306    1329
AMTYP2= 000000      1306    1337
AMTYP3= 000000      1306    1340
AMTYP4= 000000      1306    1343
APASS = 000000      1306    1311
APRIOR= 000005      1052#   1306
APTCSU= 000040      7055#   7236
APTENV= 000001      7011    7053#   7087    7229
APTSIZ= 000200      1805    7052#
APTSPO= 000100      7013    7054#   7231
ASWREG= 000000      1306    1319
ATESTN= 000000      1306    1310
AUNIT = 000000      1306    1313
AUSWR = 000000      1306    1320
AVECT1= 120210      1051#   1306    1345
AVECT2= 000000      1306    1346
BADPAR  002334      1739#   6782*
BAI   = 000020      1111#
```

```
BA16   = 000400     1096#
BA17   = 001000     1097#
BITCNT   002326     1736#   2930*   2947*   3060*   3077*   3193*   3210*   3324*   3341*   3455*   3472*   3586*   3603*
                    3718*   3735*   4041*   4058*   4172*   4189*   4634*   4651*   5458*   5483*   5496*   5511*   5554*
                    5613*   5739*   5744*   5746*   5759*   5783*   5793*   5804*   5841*   5873*   5916*   5942*   6017*
                    6022*   6024*   6037*   6061*   6072*   6084*   6096*   6183*   6212    6220*   6239    6265    6273*
                    6274    6277    6290    6295    7874    7877    7883    7886    7893    7896
BIT0   = 000001      997#   1093    1126    1145    3286    3416    3547    4133    4909    5006    5585    5590    5855
                    5904    5930    6197    6253    6323    6761
BIT00  = 000001      987#    997
BIT01  = 000002      986#    996
BIT02  = 000004      985#    995
BIT03  = 000010      984#    994
BIT04  = 000020      983#    993
BIT05  = 000040      982#    992
BIT06  = 000100      981#    991
BIT07  = 000200      980#    990
BIT08  = 000400      979#    989    6915
BIT09  = 001000      978#    988    6933    7098
BIT1   = 000002      996#   1127    3417    4134    5360    6394
BIT10  = 002000      977#   1098    1117    1136    1169    1184    6418    6420    7075
BIT11  = 004000      976#   1099    1118    1137    1154    1170    1185    3283    3413    3544    4130    6940
BIT12  = 010000      975#   1101    1119    1138    1171    3284    3414    3545    3548    4131    5660
BIT13  = 020000      974#   1102    1120    1139    1155    1172    6414    6416    7082
BIT14  = 040000      973#   1103    1121    1140    1156    1173    3284    3414    3545    4131    5564    5894    6230
                    6901
BIT15  = 100000      972#   1104    1105    1122    1141    1157    1174    4395    5564    5634    6762
BIT2   = 000004      995#   1128    1147
BIT3   = 000010      994#   1110    1129    1148
BIT4   = 000020      993#   1111    1130    1149    1163    1178    6414    6416
BIT5   = 000040      992#   1112    1131    1150    1164    1179
BIT6   = 000100      991#   1094    1113    1132    1151    1165    1180    2887    2889    2891    3017    3019    3021
                    3150    3152    3154    3284    3414    3545    3675    3677    3679    3998    4000    4002    4131
                    4590    4592    4594    5210    5212    5214    5654
BIT7   = 000200      990#   1095    1114    1133    1152    1166    1181
BIT8   = 000400      989#   1096    1115    1134    1153    1167    1182    5658    6418    6420
BIT9   = 001000      988#   1097    1116    1135    1168    1183    1980    2098    2215    2332    2453    3283    3413
                    3544    4130    5388    5400    5409    5417    5425    5430    5473    5520    5568    5599    5651
                    5703    5765    5814    5968    5978    6043    6106    6355    6372
BLDDAT   032014     1926    2044    2161    2278    2399    2514    2597    2685    2792    2879    3009    3142    3275
                    3405    3536    3667    3798    3903    3990    4122    4254    4349    4483    4576    4712    4803
                    5196    6619#
BPTVEC= 000014      1004#
BSE    = 000200     1133#   4417    4419
BSEDES   002352     1746#   1914*   2034*   2151*   2268*   2386*   2503*   2586*   2670*   2777*   2865*   2995*   3128*
                    3260*   3391*   3522*   3653*   3783*   3888*   3976*   4108*   4239*   4336*   4395*   4411*   4469*
                    4564*   4699*   4790*   4895*   4992*   5085*   5185*   5574    5576    5577
CCLR   = 100000     1105#   1912    2032    2149    2266    2384    2504    2587    2671    2778    2866    2996    3129
                    3262    3392    3523    3654    3784    3889    3977    4109    4240    4334    4493    4597    4722
                    4813    4914    5011    5099    5217
CDT    = 002000     1098#
CERR   = 100000     1104#   2557    2643    2967    3097    3230    3361    3492    3623    3755    4078    4209    4439
                    4671
CFMT   = 010000     1101#   1978    2096    2213    2330    2397    2451    3794    3899    3986    4118    4708    5093
                    5366    5386    5566    5648    5677    5879    5890    6187    6226
CKSWR  = 104407     6900    7071    7097    7855#
```

```
CLEAR  = 000005       1078#
CLRIBF   031770       2683    2790    2877    3007    3140    3273    3403    3534    3665    3796    3901    3988    4120
                      4252    4347    4481    4574    4710    4801    5194    6608#
COE    = 001000       1135#   2560    2640
CR     = 000015        912#   7275    7285
CRLF   = 000200        913#   1830    7246    7285
CTO    = 004000       1099#
CTRHLT   034556       7192#   7471    7554
DCK    = 100000       1141#   2975    3105    3238    3369    3500    3631    3763    4086    4217    4679
DDISP  = 177570        919#   1264    1793
DDT    = 000400       1153#
DF000    040026       1370    7909#
DF0004   040032       1385    1487    1577    1637    1643    7911#
DF0005   040062       1391    7923#
DF0006   040106       1397    1475    7933#
DF0007   040132       1403    1481    7943#
DF0010   040156       1409    1421    1439    1517    1529    1619    7953#
DF0011   040202       1415    1427    1445    1457    1469    1523    1541    1631    7963#
DF0014   040232       1433    1595    7975#
DF0017   040256       1451    7985#
DF0021   040302       1463    1535    1625    7995#
DF0026   040326       1493    1499    1505    1511    8005#
DF0027   040346       8013#
DF0037   040362       1547    8019#
DF0040   040402       1553    8027#
DF0041   040406       1559    8029#
DF0042   040426       1565    8037#
DF0043   040446       1571    8045#
DF0045   040466       1583    1607    8053#
DF0046   040512       1589    1613    8063#
DF0050   040532       1601    8071#
DF0060   040556       1649    8081#
DF0061   040576       1655    8089#
DF0062   040622       1661    8099#
DF0063   040646       1667    8109#
DHDCMP   024276       1943    2009    2061    2127    2178    2244    2295    2361    2416    2482    2531    2614    2702
                      2809    2907    3037    3170    3301    3432    3563    3695    3815    3920    4018    4149    4271
                      4366    4403    4507    4611    4736    4827    4928    5025    5118    5231    5549#
DH000A   041232       1497    1545    1557    1563    1569    1587    1611    1647    7913    7925    7935    7945    7955
                      7965    7977    7987    7997    8007    8055    8073    8091    8101    8111    8163#
DH000B   041250       7915    7927    7937    7947    7957    7967    7979    7989    7999    8009    8015    8021    8031
                      8039    8047    8057    8065    8075    8083    8093    8103    8113    8166#
DH000C   041263       1368    8168#
DH0017   041773       7991    8233#
DH0021   042145       8001    8041    8049    8252#
DH0026   042245       8011    8017    8264#
DH004    041314       7917    7929    7939    7949    8059    8067    8077    8095    8105    8173#
DH004A   041333       7919    8079    8176#
DH004B   041351       7931    8179#
DH004C   041367       7941    8182#
DH004D   041405       7951    8185#
DH004E   041423       7921    7973    8188#
DH004F   041450       8061    8192#
DH004G   041465       8069    8097    8195#
DH004H   041502       8107    8198#
DH0041   042406       8033    8281#
```

```
DH010   041517      7959   7969   8201#
DH010A  041563      7961   8208#
DH010B  041631      7971   8215#
DH014   041677      7981   8222#
DH014A  041736      7983   8228#
DH017A  042057      7993   8242#
DH021A  042171      8003   8043   8256#
DH021B  042217      8051   8260#
DH037A  042332      8023   8273#
DH037B  042357      8025   8277#
DH041A  042452      8035   8287#
DH060A  042520      8085   8294#
DH060B  042545      8087   8298#
DH063A  042572      8115   8302#
DH063B  042637      8117   8309#
DI    = 040000      1103#
DISEEK  023334      1931   2049   2166   2283   2404   2519   2602   2690   2797   2895   3025   3158   3289
                    3420   3551   3683   3803   3908   4006   4137   4259   4354   4495   4599   4724   4815
                    4916   5013   5106   5219   5357#
DISPLA  001142      1264#  1793*  1801*  6955*  7074*
DISPRE  000174      1193#  1801
DLT   = 100000      1122#
DMD   = 000040      1164#  1962   1963   2080   2081   2197   2198   2314   2315   2435   2436   2551   2552
                    2634   2635   2834   2835   3945   3946   4302   4303   4390   4391   4532   4533   4761
                    4762   4858   4859   4951   4952   5048   5049   5141   5142   5255   5256   5335   5336
                    5371   5372   5393   5463   5488   5583   5750   5788   6028   6062   6440   6470   6484
                    6499   6525   6537   6549   6577   6580   6581   6582   6583   6586   6587   6588   6589
                    6696
DRA   = 000001      1145#
DRDY  = 000200      1152#
DREAD   027176      2715   2822   2920   3050   3183   3314   3445   3576   3708   3828   3933   4031   4162
                    4284   4379   6165#
DRGPSN  026500      2707   2814   2912   3042   3175   3306   3437   3568   3700   3820   3925   4023   4154
                    4276   4371   4512   4616   4741   4832   4933   5030   5123   5236   6009#
DROT  = 000040      1150#
DRPAR = 000010      1129#
DRSYNC  023740      1939   2005   2057   2123   2174   2240   2291   2357   2412   2478   2527   2610   2698
                    2805   2903   3033   3166   3297   3428   3559   3691   3811   3916   4014   4145   4267
                    4362   4399   4503   4607   4732   4823   4924   5021   5114   5227   5456#
DRVMSK= 000007      1109#
DSC   = 040000      1156#
DSECTR  023316      1915   1937   2003   2055   2121   2172   2238   2289   2355   2410   2476   2525   2608
                    2696   2803   2901   3031   3164   3295   3426   3557   3689   3809   3914   4012   4143
                    4265   4360   4397   4501   4605   4730   4821   4922   5019   5112   5225   5335#
DSWR  = 177570       918#  1263   1792
DTE   = 010000      1138#
DTYE  = 000040      1131#
DT000   037510      1369   7864#
DT0004  037514      1384   1486   1576   1600   1636   1642   7865#
DT0005  037532      1390   7868#
DT0006  037542      1396   1474   7870#
DT0007  037552      1402   1480   7872#
DT0010  037562      1408   1420   1438   1516   1528   1618   7874#
DT0011  037600      1414   1426   1444   1456   1468   1522   1540   1630   7877#
DT0014  037624      1432   1594   7881#
DT0017  037640      1450   7883#
```

```
DT0021   037662     1462   1534   1564   1624   7886#
DT0026   037674     1492   1498   1504   1510   7888#
DT0037   037716     1546   1648   7891#
DT0040   037722     1552   7892#
DT0041   037730     1558   7893#
DT0043   037746     1570   7896#
DT0045   037760     1582   1606   7898#
DT0046   037770     1588   1612   1654   7900#
DT0062   040000     1660   7902#
DT0063   040010     1666   7904#
DWGPSN   025204     1949   2067   2184   2301   2422   2537   2620   5731#
DWRITE   025604     1956   2074   2191   2308   2429   2544   2627   5836#
DWRTCK   027176     4520   4624   4749   4840   4941   5038   5131   5244   6164#
ECCGEN   030462     2936   3066   3199   3330   3461   3592   3724   4047   4178   4640   5867   6214   6245
                    6267   6392#
ECCHI    002340     1741#  5843*  5903   6184*  6249   6400   6406   6411*  6419   6420*  6421*  6422
ECCLO    002342     1742#  5842*  5931   6185*  6250   6410*  6415   6416*  6417*
ECCSRC   002344     1743#  2673*  2779*  2780*  2886*  3016*  3149*  3282*  3412*  3543*  3674*  3786*  3890*
                    3891*  3997*  4129*  4242*  4337*  4471*  4472*  4589*  4701*  4792*  4897*  4994*  5086*
                    5209*  6247
ECCW   = 020000     1172#  5393   5463   5488   5583   5750   5788   5886   5924   6028   6062   6276   6279
ECCXOR   002336     1740#  6402*  6408*  6412
ECH    = 000100     1132#  2973   3103   3236   3367   3498   3500   3629   3631   3761   4084   4215   4217
                    4677
ECPATX   002346     1744#  2713*  2820*  2918*  3048*  3181*  3312*  3443*  3574*  3706*  3826*  3931*  4029*
                    4160*  4282*  4377*  4518*  4622*  4747*  4838*  4939*  5036*  5129*  5242*  6292
ECPOSX   002350     1745#  2714*  2821*  2919*  2939*  2940   2952*  2960*  3049*  3069*  3070   3082*  3090*
                    3182*  3202*  3203   3215*  3223*  3313*  3333*  3334   3346*  3354*  3444*  3464*  3465
                    3477*  3485*  3575*  3595*  3596   3608*  3616*  3707*  3727*  3728   3740*  3748*  3827*
                    3932*  4030*  4050*  4051   4063*  4071*  4161*  4181*  4182   4194*  4202*  4283*  4378*
                    4519*  4623*  4643*  4644   4656*  4664*  4748*  4839*  4940*  5037*  5130*  5243*  6284
                    6297*  6300
EH026    001602     1491#  2648*
EH030    001622     1503#  2562*
EH031    001632     1509#  2566*
EMTVEC=  000030     1007#  1776*  1777*
EMW      001510     1448#  1948*  1955*  2066*  2073*  2183*  2190*  2300*  2307*  2421*  2428*  2536*  2543*
                    2619*  2626*  6445*  6475*  6489*  6504*  6530*  6542*  6554*
EMW1     043644     6445   8407#
EMW2     043732     6475   6530   8416#
EMW3     044022     6489   6542   8426#
EMW4     044110     6504   6554   8435#
EM000    042704     1367   8318#
EM5004   042751     1382   1388   1394   1400   1586   8325#
EM5010   043163     1406   1412   1442   8353#
EM5011   043243     1418   1424   1430   8362#
EM5015   043346     1436   8374#
EM5020   043403     1454   1948   2066   2183   2300   2421   2536   2619   8379#
EM5021   043462     1460   1466   1955   2073   2190   2307   2428   2543   2626   8387#
EM5023   043560     1472   1478   1484   8398#
EM5026   044200     1490   8445#
EM5027   044340     1496   8462#
EM5030   044512     1502   8480#
EM5031   044605     1508   8490#
EM5032   044672     1514   1520   8499#
EM5034   044750     1526   1538   1634   8507#
```

```
EM5035   045015           1532#   8514#
EM5037   045056           1544    8520#
EM5041   045132           1556    8528#
EM5042   045217           1562    8537#
EM5043   045300           1568    8546#
EM5044   045357           1574    1580    8554#
EM5046   045427           8561#
EM5047   045521           1592    8571#
EM5050   045614           1598    8581#
EM5051   045667           1604    8589#
EM5052   045744           1610    8597#
EM5053   046064           1616    1628    1640    8611#
EM5054   046153           1622    8621#
EM5060   046216           1646    8627#
EM5061   046275           1652    1658    8635#
EM5063   046352           1664    8643#
ERRCNT   002314           1731#   1809*   7175*   7178
ERRVEC=  000004           1000#   1790    1791*   1802*   6780*   6781*   6801*   6802*   6828    6829    6833*   6838*   6850*
                          6855*   6864*   6874*   6875*   6906    6907*   6909*   6912*
E.ASOF   002236           1696#
E.BA     002224           1691#   4298*   4445*   4446    4854*   5269*   5270*   5271    7900
E.CS1    002220           1689#   1987*   1988    2105*   2106    2222*   2223    2339*   2340    2460*   2461    2841*   2842*
                          2843    2966*   2967*   2968*   2969    3096*   3097*   3098*   3099    3229*   3230*   3231*   3232
                          3360*   3361*   3362*   3363    3491*   3492*   3493*   3494    3622*   3623*   3624*   3625    3754*
                          3755*   3756*   3757    3951*   3952*   3953*   3954    4077*   4078*   4079*   4080    4208*   4209*
                          4210*   4211    4308*   4309*   4310*   4311    4424*   4425    4438*   4439*   4440    4538*   4539*
                          4540*   4541    4670*   4671*   4672*   4673    4767*   4768*   4769*   4770    4864*   4865*   4866*
                          4867    5392*   5396    5513*   5514    5668*   5669    5777*   577P    5808    5949*   5951    6055*
                          6056    6100    6328*   6329    7865    7877
E.CS2    002230           1693#
E.DA     002226           1692#   4413*   5674*   5675*   5680    5682*   5683*   5685    5688*   5693    5698    7881
E.DB     002242           1698#
E.DCYL   002240           1697#   4412*   5673*   5689*   5691    5697    7881
E.DS     002232           1694#
E.ECPS   002252           1702#   2940*   2944    3070*   3074    3203*   3207    3334*   3338    3465*   3469    3596*   3600
                          3728*   3732    4051*   4055    4182*   4186    4644*   4648    6284*   6287    6300*   6304    7893
                          7896
E.ECPT   002254           1703#   2941    3071    3204    3335    3466    3597    3729    4052    4183    4645    5869    6216
                          6269    6285    6301    6422*   6423*   7886    7893
E.ER     002234           1695#   2975*   3105*   3238*   3369*   3500*   3631*   3763*   4086*   4217*   4419*   4679*   7898
E.MR1    002244           1699#   5393*   5407    5463*   5467    5488*   5492    5504    5583*   5593    5662*   5750*   5755
                          5788*   5886*   5924*   6028*   6033    6062*   6067    6076*   6079    6090*   6091    6208    6262
                          6276*   6279*   6280*   6439*   6442    6453*   6454*   6459*   6464*   6469*   6472    6483*   6486
                          6497*   6498*   6501    6518*   6521*   6524*   6527    6536*   6539    6548*   6551    7868    7874
                          7883
E.MR2    002246           1700#   1969*   1993    2087*   2111    2204*   2228    2321*   2345    2442*   2466    5378*   5415
                          7870
E.MR3    002250           1701#   1983*   1998    2101*   2116    2218*   2233    2335*   2350    2456*   2471    5390*   5423
                          7872
E.SPAR   002256           1704#
E.WC     002222           1690#   5276*   5277    7902
E5004A   043015           1383    1413    1425    1443    1455    1467    1485    1521    1539    1575    1599    1629    1635
                          1641    8332#
E5004B   043034           1389    1407    1419    1437    1515    1527    1617    8335#
E5004C   043053           1395    1473    8338#
E5004D   043072           1401    1479    8341#
```

```
E5004E   043111     1581   1605   8344#
E5004F   043127     1653   8347#
E5004G   043145     1659   1665   8350#
E5011A   043316     1431   1593   8370#
E5021A   043530     1461   1533   1623   8394#
E5026A   044253     2562   2566   2648   8453#
E5026B   044424     8471#
FMTE   = 000020     1130#
FSBLVV   032444     1968   1982   2086   2100   2203   2217   2320   2334   2441   2455   5377   5389   6756#
GETREG   032356     2556   2639   2840   2965   3095   3228   3359   3490   3621   3753   3950   4076   4207
                    4307   4423   4437   4537   4669   4766   4863   4956   5053   5146   5260   5394   5510
                    5666   5775   5806   5946   6053   6098   6327   6720#
GNS    = ****** U   1192   1829   5306   5313   7847   7848   7849   7850   7851   7853   7855   7856   7857
                    7858   7859   7860   7861
GO     = 000001     1093#  2841   2968   3098   3231   3362   3493   3624   3756   3952   4079   4210   4309
                    4438   4539   4672   4768   4865
GTSWR  = 104406     1824   7853#
HIBITS   002354     1747#  2674*  2781*  2868*  2998*  3131*  3264*  3394*  3525*  3656*  3787*  3892*  3979*
                    4111*  4243*  4338*  4468*  4565*  4700*  4791*  4896*  4993*  5095*  5153   5162   5164*
                    5184*  6231
HT     = 000011      910#  7244   7285
HVRC   = 000400     1134#
IBUFF    046632     2679   2726   2786   2873   3003   3136   3269   3399   3530   3661   3792   3839   3897
                    3984   4116   4248   4295   4298   4343   4445   4477   4487   4570   4580   4706   4716
                    4797   4807   4851   4854   4902   4908*  4909   4960   4966*  4999   5005*  5006   5057
                    5063*  5064*  5091   5096*  5150   5190   5200   5264   5269   6608   8739#
IDAE   = 002000     1136#
IE     = 000100     1094#
ILF    = 000001     1126#
INTR   = 000300     1089#
IOTVEC = 000020     1005#  1774*  1775*
IR     = 000100     1113#
KIPAR0 = 172340     1038#  6814   6842
KIPAR1 = 172342     1039#
KIPAR2 = 172344     1040#
KIPAR3 = 172346     1041#
KIPAR4 = 172350     1042#
KIPAR5 = 172352     1043#
KIPAR6 = 172354     1044#
KIPAR7 = 172356     1045#  6858
KIPDR0 = 172300     1027#
KIPDR1 = 172302     1028#
KIPDR2 = 172304     1029#
KIPDR3 = 172306     1030#
KIPDR4 = 172310     1031#
KIPDR5 = 172312     1032#
KIPDR6 = 172314     1033#
KIPDR7 = 172316     1034#
LF     = 000012      911#  7279   7285
LOADRK   032230     1918   2036   2153   2270   2391   2506   2589   2675   2782   2869   2999   3132   3265
                    3395   3526   3657   3788   3893   3980   4112   4244   4339   4473   4566   4702   4793
                    4898   4995   5087   5186   6692#
L.ASOF   002272     1714#  6697*  6712
L.BA     002264     1709#  6694*  6707
L.CS1    002260     1707#  1978   1987   2096   2105   2213   2222   2330   2339   2451   2460   2966   3096
                    3229   3360   3491   3622   3754   3951   4077   4208   4308   4424   4538   4670   4767
```

```
                        4864    5360    5366    5386    5392    5513    5566    5648    5668    5677    5777    5879    5890
                        5949    6055    6187    6226    6328    6394    6699*   6713
L.CS2     002270        1713#   6698*   6710
L.DA      002266        1710#   4413    5674    5698*   5736    6014    6693*   6709
L.DCYL    002274        1715#   1967    2085    2202    2319    2440    4412    5376    5565    5673    5697*   6692*   6708
L.DS      002266        1711#   1976    2094    2211    2328    2449    5385    5563    5652    5656    5741    6019
L.DT      002267        1712#   1970    2088    2205    2322    2443    5379    5556
L.MR1     002276        1716#   6696*   6711
L.MR2     002300        1717#
L.MR3     002302        1718#
L.WC      002262        1708#   6695*   6706
MCLK    = 000400        1167#   1962    2080    2197    2314    2435    2551    2634    2834    3945    4302    4390    4532
                        4761    4858    4951    5048    5141    5255    5371    6439    6440    6453    6483    6484    6498
                        6521    6536    6537    6548    6577    6580    6582    6586    6588
MDS     = 001000        1116#
MEMBAS= 172100          1048#   6784
MEMERR    031726        1211    6596#
MEMPAR    002332        1738#   6783*   6789*   6803
MEMVEC= 000114          1047#   1210    6805*   6806*
MERD    = 001000        1168#   6580    6588
MESMSK= 000017          1161#
MEWD    = 002000        1169#   5393    5463    5488    5583    5750    5788    6028    6062    6439    6454    6459    6483
                        6497    6548
MIND    = 000200        1166#
MMVEC   = 000250        1016#
MSP     = 000100        1165#   5335
M1.BIT    002322        1734#   4432*   5462*   5500*   5586*   5752*   5784*   5858    5859*   5906    5907*   5936    5937*
                        6030*   6200*   6244*   6255*   6396    6457    6516    6522    6575    7874    7877    7883
M2.BIT    002324        1735#   5785*   5858*   5906*   5936*   6462    6467    7883
NED     = 010000        1119#
NEM     = 004000        1118#
NEWPAS    003364        1886#   5329
NXF     = 000004        1128#
OBUFF     047636        1922    2040    2157    2274    2395    2510    2593    2725    2884*   2885*   2887    2889*   2891*
                        3014*   3015*   3017    3019*   3021*   3147*   3148*   3150    3152*   3154*   3280*   3281*   3283*
                        3284*   3286*   3410*   3411*   3413*   3414*   3416*   3417*   3541*   3542*   3544*   3545*   3547*
                        3548*   3672*   3673*   3675    3677*   3679*   3838    3995*   3996*   3998    4000*   4002*   4127*
                        4128*   4130*   4131*   4133*   4134*   4488    4581    4587*   4588*   4590    4592*   4594*   4717
                        4808    4907*   4911*   4961    5004*   5008*   5058    5097*   5151    5201    5207*   5208*   5210
                        5212*   5214*   5265    5845    6182    6249*   6250*   6624    6664    8740#
OFFSET= 000015          1082#
OFST    = 000004        1147#
OPI     = 020000        1139#
OPR001    040672        1836    8121#
OPR002    040724        1839    1849    1866    8126#
OPR003    040732        1846    8127#
OPR004    040762        1857    8131#
OPR006    041004        1833    8134#
OPR007    041053        7197    8141#
OPSTRT    032274        1929    2047    2164    2281    2402    2517    2600    2689    2796    2894    3024    3157    3288
                        3419    3550    3682    3802    3907    4005    4136    4258    4353    4494    4598    4723    4814
                        4915    5012    5105    5218    6706#
OR      = 000200        1114#
PACK    = 000003        1077#
PARCHK    032516        1888    6780#   7808
PARM      002360        1199    1751#
```

D14


```
PAR.EN= 000001     1050#   6787
PAT   = 000020     1163#
PAT2    046432     6658    8657#
PAT3    046472     6656    8677#
PAT4    046532     6652    8697#
PAT6    046572     6641    8719#
PCA   = 004000     1170#   6453    6469    6518
PCD   = 010000     1171#   6453    6464    6524
PGE   = 002000     1117#
PIP   = 020000     1155#
PIRQ  = 177772      917#
PIRQVE= 000240     1011#
PR.BIT  002320     1733#   4431*   5461*   5500    5501*   5586    5587*   5590*   5751*   5787*   5859    5860*   5907
                   5908*   5937    5938*   6029*   6088*   6200    6201*   6204*   6244    6255    6256*   6259*   6455
                   6573    7874    7877    7883
PR0   = 000000      934#   1889
PR1   = 000040      935#
PR2   = 000100      936#
PR3   = 000140      937#
PR4   = 000200      938#
PR5   = 000240      939#   1725
PR6   = 000300      940#
PR7   = 000340      941#   1212    1761    6781    7791    7805
PS    = 177776      914#    915
PSW   = 177776      915#
PWRVEC= 000024     1006#   1780*   1781*   7790*   7791*   7804*   7805*
P1.BIT  002316     1732#   5786*   5801*   5860    5861*   5864*   5908    5909*   5912*   5938    5939*   6393    6460
                   6514    7883
RDBIT   031616     2935    2964    3065    3094    3198    3227    3329    3358    3460    3489    3591    3620    3723
                   3752    4046    4075    4177    4206    4433    4639    4668    5465    5490    5502    5591    5753
                   6031    6065    6077    6089    6206    6260    6573#
RDCHR = 104410     7630    7856#
RDDATA= 000021     1084#   2681    2788    2875    3005    3138    3271    3401    3532    3663    3794    3899    3986
                   4118    4250    4345
RDGATE= 100000     1174#   5488    5583    5662    6076    6280
RDHEAD= 000025     1086#
RDLIN = 104411     7704    7857#
RDOCT = 104412     1840    1850    1867    7858#
RDY   = 000200     1095#   2842    2967    3097    3230    3361    3492    3623    3755    3953    4078    4209    4310
                   4439    4540    4671    4769    4866
RECAL = 000013     1081#
RESREG= 104414     7174    7860#
RESTRT  002370     1197    1754#
RESVEC= 000010     1001#
RKASOF= 000016     1064#   6712*
RKBA  = 000004     1059#   6707*
RKCS1 = 000000     1057#   1912*   1986    2032*   2104    2149*   2221    2266*   2338    2384*   2459    2504*   2587*
                   2671*   2778*   2839    2866*   2996*   3129*   3262*   3392*   3523*   3654*   3784*   3889*   3977*
                   4109*   4240*   4334*   4493*   4597*   4722*   4813*   4914*   5011*   5099*   5217*   6713*
RKCS2 = 000010     1061#   6710*   7193*
RKDA  = 000006     1060#   6709*
RKDB  = 000024     1066#
RKDCYL= 000020     1065#   6708*
RKDS  = 000012     1062#
RKECPS= 000030     1070#   2938    3068    3201    3332    3463    3594    3726    4049    4180    4642    6283    6299
RKECPT= 000032     1071#   2937    3067    3200    3331    3462    3593    3725    4048    4179    4641    5868    [illegible]
```

```
                      6268     6282     6298
RKER   = 000014       1063#    4416
RKMR1  = 000026       1067#    1962*    1963*    2080*    2081*    2197*    2198*    2314*    2315*    2435*    2436*    2551*    2552*
                      2634*    2635*    2834*    2835*    3945*    3946*    4302*    4303*    4390*    4391*    4532*    4533*    4761*
                      4762*    4858*    4859*    4951*    4952*    5048*    5049*    5141*    5142*    5255*    5256*    5335*    5336*
                      5371*    5372*    5466     5491     5503     5592     5754     6032     6066     6078     6090     6207     6261
                      6440*    6441     6470*    6471     6464*    6485     6499*    6500     6525*    6526     6537*    6538     6549*
                      6550     6577*    6580*    6581*    6582*    6583*    6586*    6587*    6588*    6589*    6711*
RKMR2  = 000034       1068#    1984     2102     2219     2336     2457
RKMR3  = 000036       1069#    1985     2103     2220     2337     2458
RKPRI    002306       1725#    1883*
RKSPAR= 000022        1072#
RKVEC    002304       1724#    1881*    1882*
RKWC   = 000002       1058#    6706*
RLS    = 000010       1110#
SAVREG= 104413        7121     7859#
SAVSWR   002356       1748#    7789*    7809
SCLR   = 000040       1112#
SCOP1  = 104415       4964     5061     5160     7861#
SCOP1$   033566       6993#    7861
SEEK   = 000017       1083#
SELDRV= 000001        1076#
SKI    = 000002       1127#
SPACE2   041116       7157     7169     8147#
SPAR   = 020000       1102#
SPDLSS= 000020        1149#
SRTFLG   002312       1727#    1751*    1754*    1757*    1834
SRTSPL= 000011        1080#
SR0    = 177572       1020#    6839     6854*    6861*
SR1    = 177574       1021#
SR2    = 177576       1022#
SR3    = 172516       1023#    6851*
STACK  = 001100        905#    1760     1772     5439     6564     6604     7185     7196     7806
START    002400       1196     1757#
START1   002404       1752     1755     1758#    7204
STKLMT= 177774         916#
SVAL   = 100000       1157#
SWR      001140       1263#    1770     1792*    1794     1800*    1807*    1822     5400     5409     5417     5425     5430     5473
                      5520     5599     5703     5765     5814     5968     5978     6043     6106     6355     6372     6449     6479
                      6493     6508     6534     6546     6558     6600     6901     6915     6918     6933     6940     6993     7075
                      7082     7094     7098     7176     7474     7513     7568*    7789     7809*
SWREG    000176       1194#    1800     1822     7474     7513     7536
SW0    = 000001        969#
SW00   = 000001        959#     969
SW01   = 000002        958#     968
SW02   = 000004        957#     967
SW03   = 000010        956#     966
SW04   = 000020        955#     965
SW05   = 000040        954#     964
SW06   = 000100        953#     963
SW07   = 000200        952#     962
SW08   = 000400        951#     961
SW09   = 001000        950#     960
SW1    = 000002        968#
SW10   = 002000        949#
SW11   = 004000        948#
```

```
SW12   = 010000          947#    7176
SW13   = 020000          946#
SW14   = 040000          945#
SW15   = 100000          944#
SW2    = 000004          967#
SW3    = 000010          966#
SW4    = 000020          965#
SW5    = 000040          964#
SW6    = 000100          963#
SW7    = 000200          962#
SW8    = 000400          961#
SW9    = 001000          960#    6449    6479    6493    6508    6534    6546    6558    6600    6993
S.CLR  = 000400         1182#
S.FMT  = 001000         1183#
S.PACK= 004000          1185#
S.RECL= 000040          1179#
S.RTC  = 000200         1181#
S.SEEK= 000020          1178#
S.STSP= 000100          1180#
S.UNLD= 002000          1184#
TBITVE= 000014          1002#
TKVEC  = 000060         1009#    7451*   7452*
TPVEC  = 000064         1010#
TRAPPC   002310         1726#    6597*   7864
TRAPVE= 000034          1008#    1778*   1779*
TRTVEC= 000014          1003#
TSTBY1   041206         8157#
TSTBY2   041216         8159#
TST1     003412         1890     1909#   6960
TST10    006622         2667#    6967
TST11    007136         2774#    6968
TST12    007430         2862#    6969
TST13    010156         2993#    6970
TST14    010704         3125#    6971
TST15    011432         3257#    6972
TST16    012160         3388#    6973
TST17    012714         3519#    6974
TST2     004030         1991     1996    2001    2029#   6961
TST20    013450         3650#    6975
TST21    014176         3780#    6976
TST22    014512         3885#    6977
TST23    015004         3973#    6978
TST24    015532         4105#    6979
TST25    016266         4236#    6980
TST26    016602         4331#    6981
TST27    017264         4421     4428    4443    4466#   6982
TST3     004442         2109     2114    2119    2146#   6962
TST30    017600         4561#    6983
TST31    020350         4696#    6984
TST32    020660         4787#    6985
TST33    021216         4892#    6986
TST34    021546         4989#    6987
TST35    022102         5082#    6988
TST36    022456         5181#    6989
TST4     005054         2226     2231    2236    2263#   6963
TST5     005466         2343     2348    2353    2381#   6964
```

```
TST6     006110          2464    2469    2474    2500#   6965
TST7     006360          2583#   6966
TYPDS  = 104405          5310    5317    7851#
TYPE   = 104401          1816    1833    1836    1839    1846    1849    1857    1866    5304    5311    5318    7077    7085
                         7133    7134    7138    7139    7146    7157    7159    7169    7170    7172    7180    7197    7249
                         7345    7420    7468    7480    7534    7535    7538    7549    7559    7570    7589    7637    7643
                         7648    7652    7657    7658    7660    7663    7667    7733    7735    7807    7847#
TYPERR   034260          7084    7121#
TYPOC  = 104402          1838    1865    7154    7537    7848#
TYPON  = 104404          7850#
TYPOS  = 104403          7849#
T.ASOF   002176          1677#
T.BA     002164          1672#   4294    4446    4850    5271    7900
T.CS1    002160          1670#   1986*   1988    2104*   2106    2221*   2223    2338*   2340    2459*   2461    2557    2643
                         2839*   2843    2969    3099    3232    3363    3494    3625    3757    3954    4080    4211    4311
                         4425    4440    4541    4673    4770    4867    5396    5514    5669    5778    5808    5951    6056
                         6100    6329    6725    7865    7877    7888
T.CS2    002170          1674#   4957    4959    5054    5056    5147    5149    5261    5263    7865    7877    7888
T.DA     002166          1673#   5693    7881    7888
T.DB     002202          1679#
T.DCYL   002200          1678#   5691    7881    7888
T.DS     002172          1675#   7865    7877    7888
T.ECPS   002212          1683#   2938*   2944    3068*   3074    3201*   3207    3332*   3338    3463*   3469    3594*   3600
                         3726*   3732    4049*   4055    4180*   4186    4642*   4648    6283*   6287    6299*   6304    6734*
                         7893    7896
T.ECPT   002214          1684#   2937*   2941    3067*   3071    3200*   3204    3331*   3335    3462*   3466    3593*   3597
                         3725*   3729    4048*   4052    4179*   4183    4641*   4645    5868*   5869    6215*   6216    6268*
                         6269    6282*   6285    6298*   6301    6735*   7886    7893
T.ER     002174          1676#   2560    2640    2973    3103    3236    3367    3498    3629    3761    4084    4215    4416*
                         4417    4677    7865    7877    7888    7898
T.MR1    002204          1680#   5407    5466*   5467    5491*   5492    5503*   5504    5592*   5593    5754*   5755    6032*
                         6033    6066*   6067    6078*   6079    6091    6207*   6208    6261*   6262    6441*   6442    6471*
                         6472    6485*   6486    6500*   6501    6526*   6527    6538*   6539    6550*   6551    7868    7874
                         7883
T.MR2    002206          1681#   1984*   1993    2102*   2111    2219*   2228    2336*   2345    2457*   2466    5415    6736*
                         7870
T.MR3    002210          1682#   1985*   1998    2103*   2116    2220*   2233    2337*   2350    2458*   2471    5423    6737*
                         7872
T.SPAR   002216          1685#
T.WC     002162          1671#   5277    7888    7902
UFE    = 000400          1115#
UNLOAD = 000007          1079#
UNS    = 040000          1140#
UPE    = 020000          1120#
VV     = 000100          1151#
WCE    = 040000          1121#   4957    5054    5147    5261
WLE    = 004000          1137#
WRDATA = 000023          1085#   1924    2042    2159    2276    2397    2512    2595
WRDCNT   002330          1737#
WRHEAD = 000027          1087#
WRL    = 004000          1154#
WRTBIT   030652          5791    5802    5865    5913    5940    6439#
WRTCHK = 000031          1088#   4479    4572    4708    4799    4904    5001    5093    5192
WRTGAT = 040000          1173#   5788    6454
WR.PAR = 000004          1049#
$APTHD   001000          1223    1229#
```

```
$ASTAT= ****** U        7033     7048
$ATYC    033636         7004     7006#
$ATY1    033612         7002#
$ATY3    033620         7003#    7234
$ATY4    033630         7005#    7090
$AUTOB   001134         1260#    1826*    7527     7684
$BASE    001324         1347#    1837     1845*    1886     7152     7810
$BDADR   001122         1255#
$BDDAT   001126         1257#
$BELL    001240         1298#    7077     7110     7480     7677
$CHARC   035114         7251*    7261*    7268     7277*    7282#
$CKSWR   036070         7513#    7855
$CMTAG   001100         1243#    1767     1768     1776     1782     1783     1784
$CM1   = 000010         1275#    1276#    1277#    1278#    1279#    1280#    1281#    1282#    1283#
$CM2   = 000020         1275#    1276#    1277#    1278#    1279#    1280#    1281#    1282#    1283#
$CM3   = 000010         1273#    1275
$CM4   = 000015         1283#    1284#    1285#    1286#    1287#    1288#    1289#    1290#    1291#    1292#    1293#    1294#    1295#
                        1296#
$CNTLC   036766         7468     7549     7677#
$CNTLG   037000         7534     7679#
$CNTLU   036773         7559     7652     7678#
$CORE    033060         6837     6864#
$CPUOP   001276         1321#
$CRLF    001245         1300#    5318     7085     7110     7133     7138     7146     7159     7172     7250     7285     7570     7657
                        7677     7738
$CROUT   033110         6864     6871#
$DBLK    035562         7386     7420     7428#
$DEVCT   001260         1312#
$DEVM    001326         1348#
$DOAGN   023306         5300     5321     5327#
$DTBL    035552         7389     7424#
$ENDAD   023276         1206     1814     5323#    7105
$ENDCT   023142         1782     5302#
$ENULL   023312         5330#
$ENV     001270         1317#    1820     7011     7035     7087     7229
$ENVM    001271         1318#    1805     7013     7231     7236
$EOP     023106         5292#    7186     7201
$EOPCT   023134         1782*    5299#    5303     7184*    7200*
$ERFLG   001103         1246#    1759*    6890     6929     6931     6937*    6959     6995     7072*    7110     7195*
$ERMAX   001115         1252#    1785*    6931     6954*    6959
$ERROR   034060         1776     7070#
$ERRPC   001116         1253#    7079*    7080*    7081     7110     7865     7868     7870     7872     7874     7877     7881     7883
                        7886     7888     7891     7893     7896     7898     7900     7902     7904
$ERRTB   001330         1365#    7130
$ERTTL   001112         1250#    5315     5319*    7078*    7110
$ESCAP   001236         1297#    1784*    1887*    2729*    3842*    5398*    5411*    5419*    5427*    5433*    5438*    5459*    5470*
                        5472*    5477*    5478*    5494*    5507*    5522*    5595*    5598*    5603*    5604*    5705*    5757*    5764*
                        5769*    5770*    5816*    5957*    5971*    5977*    5982*    5983*    6035*    6042*    6047*    6048*    6070*
                        6081*    6093*    6108*    6335*    6339*    6343*    6347*    6351*    6359*    6371*    6377*    6379*    6444*
                        6474*    6488*    6503*    6510*    6529*    6541*    6553*    6560*    6596*    6599*    6953*    7101     7103
                        7110     7194*
$ETABL   001270         1316#
$ETEND   001330         1235     1349#
$FATAL   001252         1309#    7039*
$FFLG    034056         7002*    7005*    7033     7042*    7050#
$FILLC   001156         1271#    7254     7285
```

```
$FILLS  001155       1270#   7285
$GDADR  001120       1254#
$GDDAT  001124       1256#
$GET42  023266       5320#
$GTSWR  036160       7535#   7853
$HD   = 000000        890
$HIBTS  001000       1230#
$HIOCT  037166       7726*   7737#
$ICNT   001104       1247#   6944*   6945    6947*   6958
$INTAG  001135       1261#   7532*   7551    7571    7684
$ITEMB  001114       1251#   7081*   7089    7110    7124
$KTNEX  033052       6838    6863#
$KTOUT  033042       6855    6860#
$KT11   032710       6836#   6840*   6863*
$LF     001246       1301#   7110    7285    7667    7677    7738
$LFLG   034055       7043*   7049#
$LPADR  001106       1248#   1786*   6926*   6935*   6951*   6956    6958    7811
$LPERR  001110       1249#   1787*   4912*   5009*   5098*   5440    6565    6605    6935    6952*   6958    6997    7100
$LSTAD  033154       6876*   6883#
$LSTBK  033156       6877*   6884#
$MADR1  001302       1334#
$MADR2  001306       1338#
$MADR3  001312       1341#
$MADR4  001316       1344#
$MAIL   001250       1231    1235    1307#   1804    1820    6950    7087    7229
$MAMS1  001300       1328#
$MAMS2  001304       1336#
$MAMS3  001310       1339#
$MAMS4  001314       1342#
$MBADR  001002       1231#
$MFLG   034054       7003*   7009    7044*   7048#
$MNEW   037016       7538    7682#
$MSGAD  001264       1314#   7019*   7022
$MSGLG  001266       1315#   7024*
$MSGTY  001250       1308#   7017    7025*   7037    7041*
$MSWR   037005       7535    7680#
$MTYP1  001301       1329#
$MTYP2  001305       1337#
$MTYP3  001311       1340#
$MTYP4  001315       1343#
$MXCNT  033470       6948    6958#
$NULL   001154       1269#   7256    7285
$NWTST= 000001       1894#   1896    2014#   2016    2132#   2134    2249#   2251    2367#   2369    2487#   2489    2571#
                     2573    2655#   2657    2762#   2764    2847#   2849    2979#   2981    3110#   3112    3243#   3245
                     3374#   3376    3505#   3507    3636#   3638    3768#   3770    3873#   3875    3958#   3960    4091#
                     4093    4224#   4226    4317#   4319    4454#   4456    4546#   4548    4684#   4686    4774#   4776
                     4873#   4875    4970#   4972    5068#   5070    5168#   5170
$OCNT   035342       7317*   7346*   7359#
$OMODE  035344       7312*   7316*   7321    7324*   7335*   7361#
$OVER   033454       6902    6927    6936    6946    6955#
$PASS   001256       1311#   1804*   5296*   5297*   5308    5330    6942    6959
$PASTM  001006       1233#
$POWER  037412       7807    7814#
$PWRCT  037406       7798*   7799*   7800*   7802*   7813#
$PWRDN  037264       1780    7789#   7804
$PWRUP  037312       7790    7798#
```

```
$QUES    001244      1299#  7110   7285   7589   7660   7677   7735   7738
$RDCHR   036432      7602#  7856
$RDDEC=  ****** U    7859
$RDLIN   036522      7625#  7857
$RDOCT   037030      7699#  7858
$RDSZ =  000010      7618#
$REGAD   001160      1273#
$REG0    001162      1275#  2745*  3858*  4959*  5056*  5149*  5263*  7892   7904
$REG1    001164      1276#  2746*  3859*  4960*  5057*  5150*  5264*  7892   7904
$REG2    001166      1277#  2747*  3860*  4961*  5058*  5151*  5265*  7892   7904
$REG3    001170      1278#  5152*  7904
$REG4    001172      1279#  5153*  5155*  7904
$REG5    001174      1280#
$REG6    001176      1281#
$REG7    001200      1282#
$RESRE   037226      7771#  7860
$RTNAD   023310      5329#
$R2A   = ****** U    7861
$SAVRE   037170      7755#  7859
$SCOPE   033160      1774   6899#
$SETUP=  000137      1751#  1773   1774   1776   1778   1780   1782   1783   1784   1786   1814   1817   5294
                     6900   7071   7097   7105   7472   7477   7478   7508   7684
$SIZE    032652      1884   6824#
$SIZEX   033114      6862   6872#
$STUP =  177777      1751#
$SVLAD   033420      6910   6949#
$SVPC =  000220      1204#  1209
$SWR  =  167400       878#   890    894    895    896    897    898    899    900    901   1296   1297   1298
                     1783   1784   1786   1787   1910   2030   2147   2264   2382   2501   2584   2668   2775
                     2863   2994   3126   3258   3389   3520   3651   3781   3886   3974   4106   4237   4332
                     4467   4562   4697   4788   4893   4990   5083   5182   5289   5295   5322   5328   5330
                     6891   6892   6893   6894   6895   6901   6913   6915   6916   6929   6930   6931   6938
                     6939   6940   6952   6955   6958   7062   7063   7064   7065   7066   7075   7082   7094
                     7098   7110
$SWREG   001272      1319#  1807
$SWRMK=  000000       901    902   6895   6896   6919
$SWOBT   033472      5477   5603   5769   5982   6047   6377   6925   6959#
$TESTN   001254      1310#  5475   5601   5767   5980   6045   6375   6950*  7122*  7123*  7864   7865   7868
                     7870   7872   7874   7877   7881   7883   7886   7888   7891   7893   7896   7898   7900
                     7902   7904
$TIMES   001234      1296#  1783*  1910*  2030*  2147*  2264*  2382*  2501*  2584*  2668*  2775*  2863*  2994*
                     3126*  3258*  3389*  3520*  3651*  3781*  3886*  3974*  4106*  4237*  4332*  4467*  4562*
                     4697*  4788*  4893*  4990*  5083*  5182*  5295*  6938*  6945   6948*  6958
$TKB     001146      1266#  7432   7453   7464   7489   7517   7544
$TKCNT   035572      7433#  7448#  7478   7495*  7609   7611*
$TKINT   035602      1765   7448#  7469   7530
$TKQEN=  035601      7437#  7503   7614
$TKQIN   035574      7434#  7449*  7450   7501*  7502*  7503   7505*
$TKQOU   035576      7435#  7450*  7612   7613*  7614   7616*
$TKQSR   035600      7436#  7449   7505   7616
$TKS     001144      1265#  7432   7454*  7485*  7487   7493*  7515   7531*  7541   7553*  7573*
$TKSRV   035652      7451   7464#
$TMP0    001202      1283#  1841*  1843   1845   1851*  1853   1856   1868*  1870   1872   1874*  1875*  1876*
                     1877*  1878*  1880
$TMP1    001204      1284#  5582*  5619   5622*  5625   5628*  5631   5635*  5638   5661*
$TMP10   001222      1291#  6179*  6346
```

```
$TMP11   001224          1292#   6181*   6350    6354
$TMP12   001226          1293#   6189*   6193*   6239    6265
$TMP13   001230          1294#   6190*   6194*   6274
$TMP14   001232          1295#   6191*   6195*   6277    6290    6295
$TMP2    001206          1285#
$TMP3    001210          1286#   6175*   6370
$TMP4    001212          1287#   6177*   6368
$TMP5    001214          1288#   2933*   2956    2961*   3063*   3086    3091*   3196*   3219    3224*   3327*   3350    3355*
                         3458*   3481    3486*   3589*   3612    3617*   3721*   3744    3749*   4044*   4067    4072*   4175*
                         4198    4203*   4637*   4660    4665*   5854*   5881    5892    5896*   5899*   5902*   5920    5927
                         5929*   6186*   6228    6232*   6234*   6246*   6313    6317*   6320    6324*
$TMP6    001216          1289#   5846*   5960    5963    5966    6171*   6334    6342
$TMP7    001220          1290#   2730*   2749    2751*   3843*   3862    3864*   5848*   5947    5956    6173*   6338
$TN    = 000037           879#    890    1894    1910#   1991    1996    2001    2014    2030#   2109    2114    2119    2132
                         2147#   2226    2231    2236    2249    2264#   2343    2348    2353    2367    2382#   2464    2469
                         2474    2487    2501#   2571    2584#   2655    2668#   2762    2775#   2847    2863#   2979    2994#
                         3110    3126#   3243    3258#   3374    3389#   3505    3520#   3636    3651#   3768    3781#   3873
                         3886#   3958    3974#   4091    4106#   4224    4237#   4317    4332#   4421    4428    4443    4454
                         4467#   4546    4562#   4684    4697#   4774    4788#   4873    4893#   4970    4990#   5068    5083#
                         5168    5182#   6920    6960
$TPB     001152          1268#   7274*   7285
$TPFLG   001157          1272#   7223    7285
$TPS     001150          1267#   7272    7285
$TRAP    037420          1778    7824#
$TRAP2   037442          7835#   7846
$TRP   = 000016          7839#   7848#   7849#   7850#   7851#   7852#   7853    7854#   7855    7856#   7857#   7858#   7859#
                         7860#   7861#   7862#
$TRPAD   037454          7829    7846#
$TSTM    001004          1232#
$TSTNM   001102          1245#   5294*   6890    6922*   6949*   6950    6955    6959    7074    7110    7122
$TTYIN   036756          7627    7628    7640    7658    7672    7676#
$TYPBN= ****** U         7852
$TYPDS   035346          7374#   7851
$TYPE    034636          7030    7223#   7839    7847
$TYPEC   035050          7253    7260    7267    7272#   7273    7575
$TYPEX   035116          7278    7280    7283#
$TYPOC   035144          7315#   7848
$TYPON   035160          7314    7317#   7850
$TYPOS   035120          7310#   7849
$UNIT    001262          1313#
$UNITM   001010          1234#
$USWR    001274          1320#
$VECT1   001320          1345#   1847    1855*   1856*   1859    1879*   1880*   1881    1883
$VECT2   001322          1346#
$XTSTR   033172          6904#
$$GET4= 000000           5322#
$$SW08= 000037           6959#   6960    6961#   6962#   6963#   6964#   6965#   6966#   6967#   6968#   6969#   6970#   6971#
                         6972#   6973#   6974#   6975#   6976#   6977#   6978#   6979#   6980#   6981#   6982#   6983#   6984#
                         6985#   6986#   6987#   6988#   6989#   6990#
$OFILL   035343          7311*   7315*   7325    7360#
$40CAT= ****** U         6901    7084
.      = 050642          1188#   1192#   1198#   1204    1205#   1207#   1209#   1210#   1213#   1219    1220#   1222#   1224#
                         1242#   1302    1771    1786    1787    1830#   5307#   5330    5331#   6958    6959    7051#   7110
                         7285    7428#   7432    7436#   7437    7438#   7676#   7677    7684#   7738    7793    8652#   8739#
                         8740#
.$ASTA= ****** U         7003    7006
```

```
.$X    = 001000          1219#    1224
```

```
BYPASS   1214#
COMMEN   1012#
ENDCOM   1012#
ERROR     906#  1932  1933  1934  1935  1940  1941  1944  1945  1946  1950  1951  1952  1953  1957
         1958   1959  1990  1995  2000  2006  2007  2010  2011  2012  2050  2051  2052  2053  2058
         2059   2062  2063  2064  2068  2069  2070  2071  2075  2076  2077  2108  2113  2118  2124
         2125   2128  2129  2130  2167  2168  2169  2170  2175  2176  2179  2180  2181  2185  2186
         2187   2188  2192  2193  2194  2225  2230  2235  2241  2242  2245  2246  2247  2284  2285
         2286   2287  2292  2293  2296  2297  2298  2302  2303  2304  2305  2309  2310  2311  2342
         2347   2352  2358  2359  2362  2363  2364  2405  2406  2407  2408  2413  2414  2417  2418
         2419   2423  2424  2425  2426  2430  2431  2432  2463  2468  2473  2479  2480  2483  2484
         2485   2520  2521  2522  2523  2528  2529  2532  2533  2534  2538  2539  2540  2541  2545
         2546   2547  2563  2567  2603  2604  2605  2606  2611  2612  2615  2616  2617  2621  2622
         2623   2624  2628  2629  2630  2645  2649  2691  2692  2693  2694  2699  2700  2703  2704
         2705   2708  2709  2710  2711  2717  2718  2719  2720  2721  2722  2752  2755  2798  2799
         2800   2801  2806  2807  2810  2811  2812  2815  2816  2817  2818  2824  2825  2826  2827
         2828   2829  2845  2896  2897  2898  2899  2904  2905  2908  2909  2910  2913  2914  2915
         2916   2922  2923  2924  2925  2926  2927  2943  2946  2971  2976  3026  3027  3028  3029
         3034   3035  3038  3039  3040  3043  3044  3045  3046  3052  3053  3054  3055  3056  3057
         3073   3076  3101  3106  3159  3160  3161  3162  3167  3168  3171  3172  3173  3176  3177
         3178   3179  3185  3186  3187  3188  3189  3190  3206  3209  3234  3239  3290  3291  3292
         3293   3298  3299  3302  3303  3304  3307  3308  3309  3310  3316  3317  3318  3319  3320
         3321   3337  3340  3365  3370  3421  3422  3423  3424  3429  3430  3433  3434  3435  3438
         3439   3440  3441  3447  3448  3449  3450  3451  3452  3468  3471  3496  3501  3552  3553
         3554   3555  3560  3561  3564  3565  3566  3569  3570  3571  3572  3578  3579  3580  3581
         3582   3583  3599  3602  3627  3632  3684  3685  3686  3687  3692  3693  3696  3697  3698
         3701   3702  3703  3704  3710  3711  3712  3713  3714  3715  3731  3734  3759  3764  3804
         3805   3806  3807  3812  3813  3816  3817  3818  3821  3822  3823  3824  3830  3831  3832
         3833   3834  3835  3865  3868  3909  3910  3911  3912  3917  3918  3921  3922  3923  3926
         3927   3928  3929  3935  3936  3937  3938  3939  3940  3956  4007  4008  4009  4010  4015
         4016   4019  4020  4021  4024  4025  4026  4027  4033  4034  4035  4036  4037  4038  4054
         4057   4082  4087  4138  4139  4140  4141  4146  4147  4150  4151  4152  4155  4156  4157
         4158   4164  4165  4166  4167  4168  4169  4185  4188  4213  4218  4260  4261  4262  4263
         4268   4269  4272  4273  4274  4277  4278  4279  4280  4286  4287  4288  4289  4290  4291
         4299   4313  4355  4356  4357  4358  4363  4364  4367  4368  4369  4372  4373  4374  4375
         4381   4382  4383  4384  4385  4386  4400  4401  4404  4405  4414  4420  4427  4442  4448
         4496   4497  4498  4499  4504  4505  4508  4509  4510  4513  4514  4515  4516  4522  4523
         4524   4525  4526  4527  4543  4600  4601  4602  4603  4608  4609  4612  4613  4614  4617
         4618   4619  4620  4626  4627  4628  4629  4630  4631  4647  4650  4675  4680  4725  4726
         4727   4728  4733  4734  4737  4738  4739  4742  4743  4744  4745  4751  4752  4753  4754
         4755   4756  4772  4816  4817  4818  4819  4824  4825  4828  4829  4830  4833  4834  4835
         4836   4842  4843  4844  4845  4846  4847  4855  4869  4917  4918  4919  4920  4925  4926
         4929   4930  4931  4934  4935  4936  4937  4943  4944  4945  4946  4947  4948  4962  5014
         5015   5016  5017  5022  5023  5026  5027  5028  5031  5032  5033  5034  5040  5041  5042
         5043   5044  5045  5059  5107  5108  5109  5110  5115  5116  5119  5120  5121  5124  5125
         5126   5127  5133  5134  5135  5136  5137  5138  5158  5220  5221  5222  5223  5228  5229
         5232   5233  5234  5237  5238  5239  5240  5246  5247  5248  5249  5250  5251  5266  5273
         5279   6598
ESCAPE   1012#
FSECWT   1719#  1918  2036  2153  2270  2391
GETPRI   1012#  6832
GETSWR   1012#  1817#
LDLPER   1214#
MSG      1894#  1896  2014# 2016  2132# 2134  2249# 2251  2367# 2369  2487# 2489  2571# 2573  2655#
         2657   2762# 2764  2847# 2849  2979# 2981  3110# 3112  3243# 3245  3374# 3376  3505# 3507
         3636#  3638  3768# 3770  3873# 3875  3958# 3960  4091# 4093  4224# 4226  4317# 4319  4454#
```

```
          4456     4546#    4548     4684#    4686     4774#    4776     4873#    4875     4970#    4972     5068#    5070     5168#    5170
MULT      1012#
NEWTST    1012#    1894     2014     2132     2249     2367     2487     2571     2655     2762     2847     2979     3110     3243     3374
          3505     3636     3768     3873     3958     4091     4224     4317     4454     4546     4684     4774     4873     4970     5068
          5168
OSECRD    1720#    2673     2780     2868     2998     3131     3264     3394     3525     3656     3786     3891     3979     4111     4242
          4337
OSECWC    1721#    4472     4566     4701     4792     4897     4994     5186
POP       1012#    5434     5479     5606     5771     5984     6049     6361     6669     6738     6772     7045     7046     7415     7727
          7776
PUSH      1012#    5357     5456     5549     5731     5836     6009     6165     6619     6720     6756     7006     7008     7029     7374
          7701     7756
REPORT    1012#
SCOPE      907#    1909     2029     2146     2263     2381     2500     2583     2667     2774     2862     2993     3125     3257     3388
          3519     3650     3780     3885     3973     4105     4236     4331     4466     4561     4696     4787     4892     4989     5082
          5181     5293
SETPRI    1012#    7605
SETTRA    7839#    7848     7849     7850     7851     7853     7855     7856     7857     7858     7859     7860     7861
SETUP     1012#    1766
SKIP      1012#    1991     1996     2001     2109     2114     2119     2226     2231     2236     2343     2348     2353     2464     2469
          2474     4421     4428     4443
SLASH     1012#
SPACE     1012#
SSECWT    1720#    2506     2589
STARS     1012#    1202     1216     1218     1225     1238     1302     1305     1894     1908     2014     2028     2132     2145     2249
          2262     2367     2380     2487     2499     2571     2582     2655     2666     2762     2773     2847     2861     2979     2992
          3110     3124     3243     3256     3374     3387     3505     3518     3636     3649     3768     3779     3873     3884     3958
          3972     4091     4104     4224     4235     4317     4330     4454     4465     4546     4560     4684     4695     4774     4786
          4873     4891     4970     4988     5068     5081     5168     5180     5285     6810     6887     6990     7001     7058     7111
          7120     7208     7287     7364     7431     7508     7523     7594     7618     7687     7740     7786     7795     7818
SWRSU     1012#    1788#
TRMTRP    7839#
TYPBIN    1012#
TYPDEC    1012#    5308     5315
TYPNAM    1012#    1810
TYPNUM    1012#
TYPOCS    1012#
TYPOCT    1012#    1837     7536
TYPTXT    1012#    5304     5311
$$CMRE    1236#    1275     1276     1277     1278     1279     1280     1281     1282
$$CMTM    1236#    1283     1284     1285     1286     1287     1288     1289     1290     1291     1292     1293     1294     1295
$$ESCA    1012#
$$NEWT    1012#    1894     2014     2132     2249     2367     2487     2571     2655     2762     2847     2979     3110     3243     3374
          3505     3636     3768     3873     3958     4091     4224     4317     4454     4546     4684     4774     4873     4970     5068
          5168
$$SET     7839#    7848     7849     7850     7851     7853     7855     7856     7857     7858     7859     7860     7861
$$SETM    1804#
$$SKIP    1012#    1991     1996     2001     2109     2114     2119     2226     2231     2236     2343     2348     2353     2464     2469
          2474     4421     4428     4443
.EQUAT     878#     902
.HEADE     878#     880
.KT11      878#    1012
.SETUP     878#    1751
.SWRHI     878#     890
.SWRLO     878#     902#
.$ACT1     878#    1200
```

```
.$APTB    1303#
.$APTH     878#    1214
.$APTY     878#    6999
.$CATC     878#    1186
.$CMTA     878#    1236
.$EOP      878#    5283
.$ERRO     878#    7056
.$ERRT     878#
.$POWE     878#
.$RDOC     878#    7685
.$READ     878#    7429
.$SAVE     878#    7738
.$SCOP     878#    6885
.$SIZE     878#    6808
.$TRAP     878#    7816
.$TYPD     878#    7362
.$TYPE     878#    7206
.$TYPO     878#    7285

. ABS.  050642      000

 ERRORS DETECTED:  0

 RM03:CZR6EB,RM03:CZR6EB.SEQ/SOL/CRF/NL:TOC/DOC=RM03:CZR6EB.P11
 RUN-TIME: 29 24 2 SECONDS
 RUN-TIME RATIO: 1392/56=24.7
 CORE USED:  30K  (59 PAGES)

 DOCUMENT PAGES:  183
```